

CONTENTS	SAFETY	1
	THE CONTROLS	2
	MOVEMENT PRINCIPLES	3
	OPERATION	4
	LOGICAL INSTRUCTION MENU	5
	POSITION INSTRUCTION MENU	6
	AUTOMATIC MENU	7
	EDITING MENU	8
	MANUAL MENU	9
	ADAPTIVITY	10
	PROGRAMMING	11
	ARC WELDING	12
	GLUING	13
	SPOT WELDING	14
	ERROR LIST	15
	INDEX	16

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

In no event shall ABB Robotics Products AB be liable for incidental or consequential damages arising from use of this document or of the software and hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Robotics Products AB's written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose. Contravention will be prosecuted.

Additional copies of this document may be obtained from ABB Robotics Products AB at its then current charge.

© ABB Robotics Products AB 1994

Article number: 3HAB 0002-2/Rev 1
Issued: M93

ABB Robotics Products AB
S-721 68 Västerås
Sweden

1 SAFETY

Before operating, servicing or in any other way handling the robot system, the Safety Manual in the Product Manual must be thoroughly studied.

Contents

Section	Page
1 Safety	
1.1 General	1:1
2 The Controls	
2.1 The Control cabinet	2:3
2.2 Mains switch	2:7
2.3 Control panel	2:8
2.3.1 AUTO mode	
2.3.2 MANUAL REDUCED SPEED mode	
2.3.3 MANUAL FULL SPEED mode	
2.3.4 Change of operation mode	
2.4 Programming unit	2:12
2.4.1 Dialog via programming unit	
2.5 Joystick	2:14
2.6 Enabling device	2:15
2.7 Monitor (option)	2:15
2.8 Program printout (option)	2:15
2.9 Computer Link (option)	2:16
2.10 Remote control	2:16
3 Movement Principles	
3.1 The Manipulator	3:3
3.2 Coordinate Systems	3:5
3.2.1 Coordinate systems for defining positions	
3.2.2 Coordinate systems in program execution	
3.2.3 Coordinate systems for manual movements	
3.2.4 Moving the robot with the joystick	
3.2.4.1 Rectangular base coordinate system	
3.2.4.2 Robot main axes coordinate system	
3.2.4.3 Robot wrist axes coordinate system	
3.2.4.4 Tool coordinate system, no basepoint active	
3.2.4.5 Tool coordinate system, basepoint active	
3.2.4.6 Reorientation around TCP	
3.3 TCP	3:18
3.3.1 Normal TCP	
3.3.1.1 Basepoint	
3.3.2 Room fixed TCP	
3.3.2.1 General	
3.3.2.2 Moving the robot with the joystick	
3.3.2.3 Program execution with fixed TCP	
3.3.2.4 Limitations	
3.4 Position programming	3:22
3.4.1 Robot configuration	
3.5 Movement velocity	3:23

Section	Page
3.6 Movement characteristics	3:25
3.6.1 Path optimization	
3.6.1.1 Path following principles	
3.6.1.2 Zones	
3.6.1.3 Velocity in corners	
3.6.1.4 Corner deviation	
3.6.1.5 Exceptions	
3.6.2 Speed optimization	
3.6.2.1 Guide text	
3.6.2.2 Programming	
3.7 Circles	3:36
3.7.1 Orientation interpolation	
3.7.2 Velocity in circle	
3.8 Soft Servo	3:38
3.8.1 Introduction	
3.8.2 Definition of softness	
3.8.3 Use of Soft Servo	
3.9 Program displacement	3:40
3.9.1 Parallel displacement with reference point (REFP)	
3.9.2 Program displacement with reference frame (FRAME)	
3.10 Singular points	3:46
4 OPERATION	
4.1 Start	4:3
4.1.1 Power on, power off, synchronization	
4.1.2 Program Execution	
4.1.3 Stop	
4.2 Restart after power failure	4:5
4.2.1 General	
4.2.2 Automatic restart	
4.2.3 Manual restart	
4.2.4 Restart from PLC	
4.3 Language selection	4:8
4.4 Software configuration	4:8
4.5 Program information	4:9
4.6 Error buffer	4:10
4.7 Disk handling	4:11
4.7.1 Initialize disk	
4.8 Computer link	4:13
4.8.1 General	
4.8.2 Commands to robot	
4.8.3 Commands from robot	
4.8.4 Spontaneous status messages	
4.8.5 Request for superior control	
4.8.6 Operation status	
4.9 Register	4:15
4.9.1 Register	
4.9.2 Position registers, location	
4.10 Inputs and outputs	4:17
4.11 Ports	4:17
4.11.1 Digital ports	
4.11.2 Analog ports	

Section	Page
4.12 SYSTEM-I/O	4:22
4.12.1 Standard SYSTEM-I/O, inputs	
4.12.2 Standard SYSTEM-I/O, outputs	
4.12.3 Additional SYSTEM-I/O at I/O MAP	
4.12.3.1 Summary of HOLD, HOLD RESET and AUTO INPUT	
4.12.3.2 Inputs	
4.12.3.3 Outputs	
4.13 Remote control panel signals, panel- I/O	4:28
5 Logical instructions menu	
5.1 GRIPPER	5:5
5.2 WAIT	5:7
5.2.1 WAIT (TIME)	
5.2.2 WAIT (INPUT)	
5.3 OUTPUT	5:9
5.4 JUMP	5:9
5.4.1 JUMP (unconditional)	
5.4.2 JUMP (conditional)	
5.4.3 JUMP (SEARCH)	
5.4.4 JUMP (NINPOS)	
5.5 VELOC	5:12
5.6 CALL	5:13
5.7 RETURN	5:15
5.8 REG	5:15
5.8.1 REG (FETCH)	
5.8.2 REG (SET)	
5.8.3 REG (TRANSFER)	
5.8.4 REG (LOC)	
5.9 TCP	5:18
5.10 INTER	5:19
5.11 COORD	5:51
5.12 GET B/ADD BLOCK	5:21
5.13 COM	5:22
5.14 SUCTRL	5:23
5.15 FRAME / FRAME DEFINE	5:24
5.16 SOFTS	5:25
5.17 EXTAX	5:26
5.18 EXTFRAME	5:27
5.19 GLUE SCA	5:27
5.20 LOAD (IRB 6000 only)	5:27
5.21 TRIM (IRB 6000 only)	5:27
5.22 LOCATE and VISCTRL	5:28

Section	Page
6 Positioning instructions menu	
6.1 V%	6:5
6.2 SAME	6:5
6.3 ZONE	6:5
6.4 SEARCH	6:6
6.4.1 SEARCH (dist)	
6.4.2 SEARCH (dir)	
6.4.3 SEARCH (auto)	
6.5 WEAVING	6:8
6.6 REFP	6:12
6.7 VCTRL	6:12
6.8 CONTOUR	6:13
6.9 CORVEC	6:14
6.10 POSPOS	6:14
6.11 STOPOS	6:16
6.12 POSLOC	6:17
6.13 CIRCLE	6:18
6.14 FRAME	6:18
6.15 TIME	6:19
6.16 PALLET	6:19
6.17 RELTOOL	6:21
6.17.1 RELTOOL (X, Y, Z)	
6.17.2 RELTOOL (via REG)	
6.18 EXTFRAME	6:23
7 Automatic menu	
7.1-4 PROGST., INS ST., BWD., SIM	7:5
7.5 DISPL	7:5
7.6 CORV ST	7:6
7.7 ALIGN	7:6
7.7.1 Tool direction	
7.7.2 External axes	
7.7.3 Homeposition	
7.8 AW REST	7:8
7.9 RESYNC	7:8
7.10 MOV REST	7:9

Section	Contents	Page
8 Editing menu		
8.1 V%		8:5
8.2 INSTNO		8:5
8.3 STEP		8:5
8.4 MODPOS		8:5
8.5 MODIFY		8:6
8.5.1 HANDCHK		
8.5.2 MODIF and MODARG		
8.5.3 DISPL		
8.5.4 MODEXT		
8.6 DELETE		8:8
8.7 INSERT		8:8
8.8 PROGRAM		8:9
8.9 RESEQ		8:9
8.10 COPY		8:10
8.11 ERASE		8:11
8.12 PROG NO		8:11
8.13 MIRROR		8:12
8.14 TIME		8:14
8.15 COPY INS		8:15
9 Manual Menu		
9.1 CLEAR		9:5
9.2 DISK		9:6
9.2.1 DISK (FROM DISK)		
9.2.2 DISK (TO DISK)		
9.2.3 DISK (INIT)		
9.2.4 DISK (DELETE)		
9.3 WINSCHESSTER MEMORY		9:8
9.3.1 WINCH (FR WINCH)		
9.3.2 WINCH (TO WINCH)		
9.3.3 WINCH (INIT)		
9.3.4 WINCH (DELETE)		
9.4 IN/OUT (INPUT, OUTPUT, REG)		9:9
9.5 TOOL (TCP, SENSOR, WRIST LOAD, SOFT S)		9:10
9.6 FRAME		9:26
9.7 LIST (prog, errors)		9:28
9.7.1 PROG		
9.7.2 ERRORS		
9.7.3 SYS PAR		
9.7.4 USER PAR		
9.8 FRSC		9:28
9.9 TO SC		9:29
9.10 RBMODE		9:30
9.11 *LANG		9:30
9.12 ERRORS		9:30
9.12.1 Software Configuration		
9.13 TEACH		9:32
9.14 WDATA (arc weld function)		9:34
9.15 PALLET		9:34
9.16 EXTFRAME		9:35
9.17 HOMEPOS (from M93/5)		9:35
9.18 REFP		9:36

Section	Contents	Page
10 Adaptivity		
10.1 Adaptivity sensor interface		10:3
10.2 Connection and definition of sensors		10:3
10.3 Searching		
10.4		
10.3.1 Distance searching		
10.3.2 Direction searching		
10.3.3 Autosearch (AW)		
10.4 Speed control		10:7
10.5 Contour tracing		10:8
10.6 Correction vector		10:10
10.7 FRAME		10:11
11 Programming		
11.1 OVERVIEW OF FIVE PROGRAMMING MENUS		11:3
11.2 Initialization sequence for the main program		11:5
11.3 PROGRAM STRUCTURE		11:5
11.4 PATTERN PROGRAM		11:7
11.5 PROGRAMMING ROBOT PATHS		11:9
11.5.1 Smooth wrist reorientation		
11.5.2 Recommendation for Arc Welding		
11.5.3 Recommendation for Glueing and Sealing		
11.5.4 Process application type plastic cutting, metal cleaning, deburring and polishing		
11.5.5 Recommendation for Assembly, Machine, Tending and Material Handling		
11.5.6 Closely spaced positions		
11.6 PROGRAMMING EXAMPLES		11:16
11.6.1 Arc welding of rear axis member		
11.6.2 Glueing of a car door		
11.6.3 Cutting and deburring of plastic, forming of holes using the CIRCLE function		
11.6.4 Cutting and deburring of plastic lining for a car door		
11.6.5 Material handling applications		
11.7 HIGH PERFORMANCE PROGRAMMING		11:23
11.7.1 General		
11.7.2 Recommendations for best performance		
11.7.3 Movement principle choice		
11.7.3.1 Coordinate system		
11.7.3.2 Movement principle		
11.8 LOAD, PROGRAMMABLE		11:27
11.8.1 General function		
11.8.2 Wrist operations (A and P).		
11.8.2.1 Acceleration index calculation.		
11.8.2.2 Trimming of position gain index.		
11.8.2.3 Preset values		
11.8.2.4 Measures to deal with overshoots.		
11.9 TRIM, PROGRAMMABLE		11:38

Section	Contents	Page
12 Arc welding		12:3
12.1 Introduction		12:3
12.2 System principles		12:3
12.3 Program structure		12:12
12.4 Description and programming of welding data		12:15
12.5 Description of robot instructions		12:61
12.5.1 Instructions for the welding process		
12.5.2 Instructions for manipulator		
12.6 Programming of robot instructions		12:64
12.6.1 Define a position with a PATH zone		
12.6.2 Instructions for the weld process		
12.6.3 Instructions for manipulator movements		
12.6.4 Common functions		
12.6.5 EXTFRAME		
12.6.5.1 Alignment of External axes		
12.7 Program execution		12:83
12.7.1 Program test		
12.7.2 Override function		
12.7.3 Supervision of welding process		
12.7.4 Execution of welding instructions		
12.8 Program example		12:88
13 Gluing		13:3
13.1 Introduction		
13.1.1 General		
13.1.2 Signals		
13.2 Programming		13:7
13.2.1 Gluing instruction		
13.2.2 Scaling instruction		
13.3 Overrides		13:15
13.3.2 Override on the scaling instruction		
13.4 Error handling		13:20
14 Spot weld		
14.1 Introduction		14:3
14.2 The standard function		14:3
14.2.1 General		
14.2.2 Welding gun, manual operation		
14.2.3 Function parameters		
14.2.4 The spot weld instruction		
14.2.5 Programming the first spot weld instruction		
14.2.6 Subsequent instructions		
14.2.7 Editing of spot weld in structions		
14.2.8 Execution of instruction without current		

Section	Contents Page
14.3 The SWI-function	14.11
14.3.1 General	
14.3.2 Function parameters	
14.3.3 I/O signals	
14.3.4 The spot weld instruction	
14.3.5 Programming the spot weld instruction	
14.3.6 Execution of instruction	
14.3.7 Repeat weld after interrupt	
14.3.8 Execution of instruction without weld current	
14.3.9 Reset of the weld controller	
14.3.10 Weld Controller programming.	
15 Error List	15:1
15.1 General	15:2
15.2 Error buffer	15:3
15.3 Operational errors	15:5
15.4 System faults	15:10
15.5 Diagnostic messages	15:35
16 Index	16:1

The Controls**Section****Page**

2.1	The Control cabinet	2:3
2.2	Mains switch	2:7
2.3	Control panel	2:8
2.3.1	AUTO mode	
2.3.2	MANUAL REDUCED SPEED mode	
2.3.3	MANUAL FULL SPEED mode	
2.3.4	Change of operation mode	
2.4	Programming unit	2:12
2.4.1	Dialog via programming unit	
2.5	Joystick	2:14
2.6	Enabling device	2:15
2.7	Monitor (option)	2:15
2.8	Program printout (option)	2:15
2.9	Computer Link (option)	2:16
2.10	Remote control	2:16

2 THE CONTROLS

2.1

The Control cabinet

External features

Control panel

Main functions for robot operation

Programming unit

All functions for robot operation and programming.
Built-in system test.

Disk drive

Storing/loading of data on diskette.

Winchester memory
(Option)

Storing/loading of data on hard disk.

Monitor
(Option)

Additional display of user program and process
messages.

Customer connections

Connectors for process communication.

Main components

Unit	Functions
Computer board	<ul style="list-style-type: none"> * Contains four microprocessors: <ul style="list-style-type: none"> - Main computer - for overall control. - Servo computer - for control of servo functions and robot movements. - Axis computer - for individual control of robot axes - I/O computer - for control of communication with operators unit, periferal equipment, host computer (option), floppy disc unit and Winchester memory. • Contains all robot primary memory.
System board	<ul style="list-style-type: none"> • Contains circuits for the personal safety functions <ul style="list-style-type: none"> - emergency stop; - work hold; - etc. * Controls, via the safety functions, power supply and brakes for the robot motors.
Digital I/O boards (option)	Digital process communication (parallel or serial).
Analogue I/O boards (option)	Analogue process communication.
Power unit	Distributes voltages in the control cabinet to drive system and robot brakes
Power supply	Stabilized voltage supply to all electronics, for the I/O-units and robot brakes.
Drive units	Current amplification and control of motor current with a common rectifier. (Up to 7 units incl. one external axis.)
Control board for external axes (option)	Reference signals for speed and connection of resolvers for 1-6 external axes (axes 7-12).
Mains transformer	Provides: <ul style="list-style-type: none"> * galvanic isolation of the control system from the mains supply; * adapts the system to mains voltages between 200 V and 600 V.
Software	Two versions: <ul style="list-style-type: none"> • Arcweld • Material handling, glue, spotweld (Shortening MH/GL/SW) <p>How to find out which version is present is described in 9.12.1.</p>

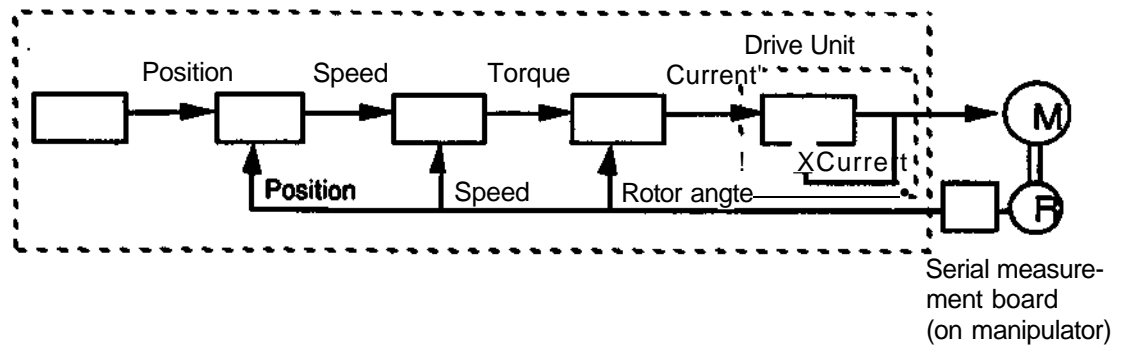
Process connections (option)

The system can handle the following input and output signals from the peripheral equipment:

- **Digital I/O**
 - up to 128 in and 128 out.
- **Analogue I/O**
 - up to 4 in and 4 out.
- **Computer communication** via RS 232 interface.
- **Control signals** for 1-6 external axes.

The AC drive system, for the robot motors and one integrated axis 7 (option), consists of the following units.

Computer board	- for overall control.
Rectifier unit	- for power supply to the drive units.
Drive unit	- receives current references from the computer board.
Serial measurement board	- monitors resolver signals

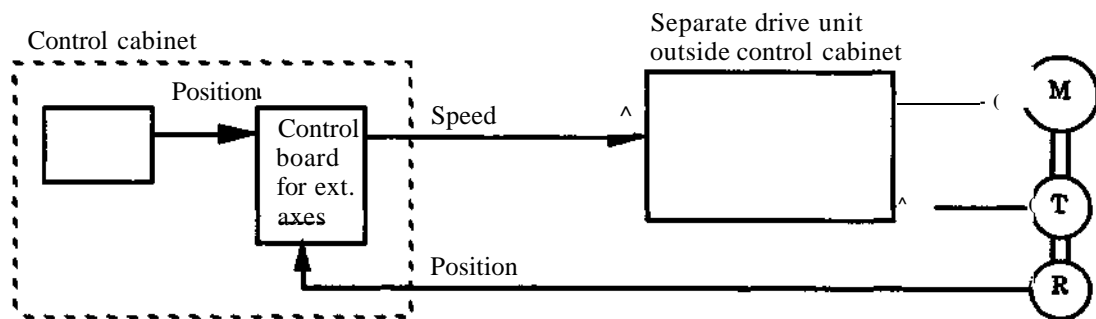


AC drive system, robot axes and integrated axis 7

2 The Controls

Drive units for optional external axes, other than the integrated axis 7, are located outside the cabinet.

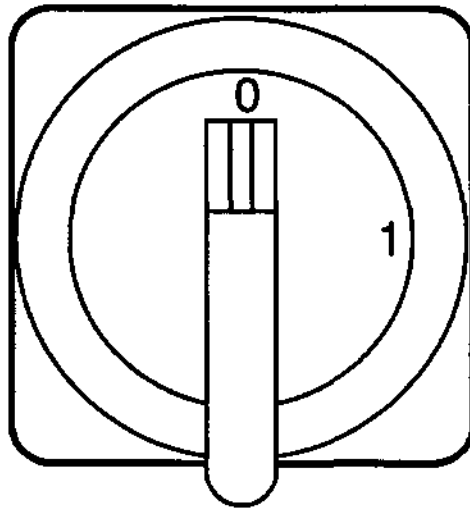
These drive units are provided with a speed reference by the control system.



Diagnostics

The control system is provided with its own built-in diagnostics with the following characteristics:

- * Test on start-up:
 - computer board
 - programming unit
 - monitor (option)
 - power supply
 - power unit
 - rectifier unit
 - location control of I/O board and drive unit
- * During test running, a red LED on the computer board is flashing.
- * Successful test running is indicated by:
 - the absence of red LEDs on all units
 - a green LED lights up on the computer board
- * Unsuccessful test running is indicated by:
 - a red LED lighting up on the faulty board
 - if it is possible, a fault message on the programming unit, (and on an optional monitor)
- * Running of the control system in a special test mode during servicing, with the following functions:
 - start-up test:
 - full test of the I/O boards, jumpers included
 - test running of the drive units at full voltage
 - program for cleaning heads on floppy disk unit

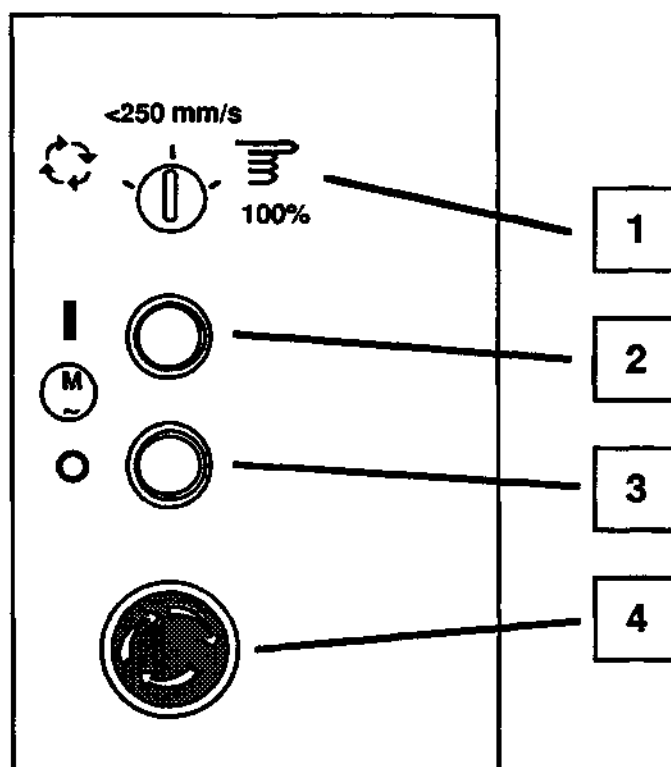
2.2**Mains switch**

The mains switch is used for connection and disconnection of the mains voltage.

If the robot is provided with the optional flange disconnect, the door will be mechanical interlocked. To open the door the flange disconnect must be in OFF position.

2.3

Control panel

**(1) Operation mode selector dockable)**

Three different operation modes may be selected:

AUTO (production mode) is used in production and program execution.

MANUAL REDUCED SPEED (programming mode) is used when working in the robot working range, when programming the robot and testing of programs.

MANUAL FULL SPEED (test mode for 100% speed) is used for testing the robot programs at full program speed.

(2) MOTOR ON

The MOTOR ON button is used in AUTO mode to put the robot to MOTOR ON mode . power is connected to the robot motors.

The MOTOR ON lamp is normally steadily lit. When a restart after a power failure is possible or when external axes are not synchronized, the MOTOR ON lamp flashes.

(3) MOTOR OFF

The MOTOR OFF button is used to put the robot in the MOTOR OFF mode regardless of the operation mode. Brakes are applied and power is disconnected from the motors.

The MOTOR OFF lamp is normally steadily lit, but flashes when an error is detected. The error indication is reset by depress of the button.

(4) EMERGENCY STOP

An emergency stop immediately stops the robot regardless of previous state and operation mode. The robot will enter MOTOR OFF mode with MOTOR OFF flashing. Emergency stop mode is reset by depress of MOTOR OFF button.

2.3.1 AUTO mode

AUTO (production mode) is used to run previously entered programs in production.

The following applies to AUTO mode:

- general mode stop is possible
- auto-mode stop is possible
- enabling device not in operation
- control via computer link/remote control is enabled
- programming/editing is disabled
- manual running (using the joystick) is disabled
- no robot speed limitation
- corrections during program execution are disabled

Some of these features can be changed by system parameters, see Installation S3

2.3.2 MANUAL REDUCED SPEED mode

MANUAL REDUCED SPEED mode (programming mode) is used when working inside the robot working range and when programming the robot and testing of programs at limited speed.

The following applies to the MANUAL REDUCED SPEED mode:

- general mode stop is possible
- auto mode stop is not possible
- enabling device in operation
- control via computer link/remote control is disabled
- programming/editing is enabled
- manual running (using the joystick) is allowed
- robot speed is limited to 250 mm/s
- HOLD TO RUN is active but can be disabled by a system parameter

2.3.3 MANUAL FULL SPEED mode

MANUAL FULL SPEED mode (test mode for 100% robot speed) is used for programming and testing the robot programs at full program speed.

The following applies to the MANUAL FULL SPEED mode:

- general mode stop is possible
- auto mode stop is not possible
- enabling device in operation
- control via computer link/remote control is disabled
- programming/editing is allowed
- manual running (using the joystick) is allowed
- no robot speed limitation
- **HOLD TO RUN** is active

2.3.4

Change of operation mode

The function "Software Key" means that the operator has to confirm the change in operation mode to AUTO mode or to MANUAL FULL SPEED mode before the new operation mode is acknowledged.

The confirmation is done by answering a check question given on the programming unit when changing the operation mode.

AUTO MODE OK?			YES	NO
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Changing from MANUAL REDUCED SPEED to AUTO

AUTO MODE				
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The operator wants to change mode, YES was selected.

PLEASE:				
CHANGE TO MANUAL REDUCED SPEED				
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

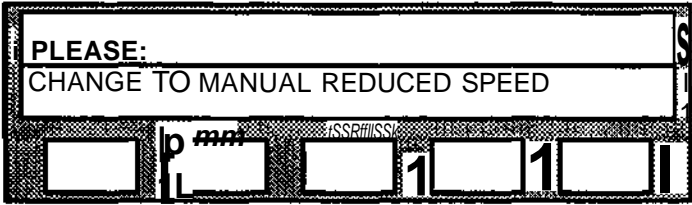
The operator does not want to change mode; NO was selected.

MANUAL FULL SPEED OK?			YES	NO
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

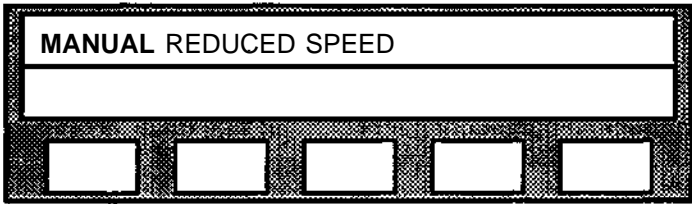
Change from MANUAL REDUCED SPEED to MANUAL FULL SPEED

MANUAL FULL SPEED				
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The operator wants to change mode, YES was selected.

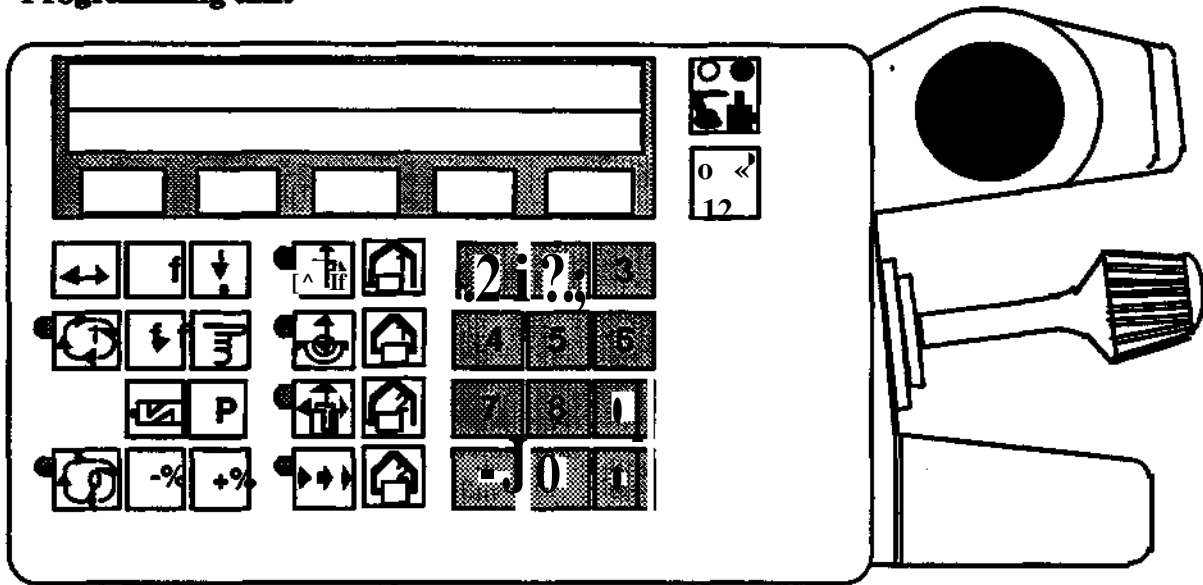


The operator does not want to change mode; NO was selected.



Change from AUTO/MANUAL FULL SPEED to MANUAL REDUCED SPEED: no question is asked.

The buttons on the programming unit are disabled until the question is answered, or until the operation mode selector is turned to the MANUAL REDUCED SPEED position.



SHIFT

- Shift of part of message on upper display bit
- Cancellation of error message



DECIMAL POINT

- Decimal point
- Shift of error message in plain language



SPEED CORRECTION

- Accuracy $\pm 5\%$ of programmed basic speed
- Current correction value shown on upper display Briefly
- Active only in MANUAL modes (can be changed by a parameter to be active also in AUTO mode)



Rectangular base coordinates



Robot coordinates



Tool coordinates



INCREMENTAL POSITIONING (toggle function)









- Fine trimming of robot position using the joystick (1 step » 0,1 mm)



GRIPPER1 OPEN, CLOSED



GRIPPER 2 OPEN, CLOSED

<u>MENUS</u>	
	LOGICAL INSTRUCTION <ul style="list-style-type: none"> • Selection of program control instructions
	POSITION <ul style="list-style-type: none"> • Selection of position instruction
	AUTOMATIC <ul style="list-style-type: none"> • Automatic program execution (continuous or step-by-step) • Simulation of input signals • Displacement of positions, programmed previously
	EDITING <ul style="list-style-type: none"> • Selection of program correction functions
	MANUAL <ul style="list-style-type: none"> • Selection of manual functions
	VISION <ul style="list-style-type: none"> • Image handling functions
	PROCESS <ul style="list-style-type: none"> • Selection of application-specific instructions
	PROGRAM STOP <ul style="list-style-type: none"> • Immediate stop of automatic program execution

2.4.1

Dialog via programming unit

Programming and other work with the robot system via the programming unit is performed as a dialogue between the operator and the robot system. The dialogue is based on a set of menus from which different alternatives can be chosen.

The menus are presented on the programming unit display. The operator selects a menu by pressing the button under the alternative required.

The robot system responds by presenting consequential questions, by presenting an information or error text or by presenting a program instruction on the display. If necessary, the menu which the operator needs to continue the dialogue is also presented.

The main menus are obtained by pressing the buttons LOGICAL INSTRUCTION, POSITION, AUTOMATIC, EDITING, MANUAL, VISION and PROCESS on the programming unit.

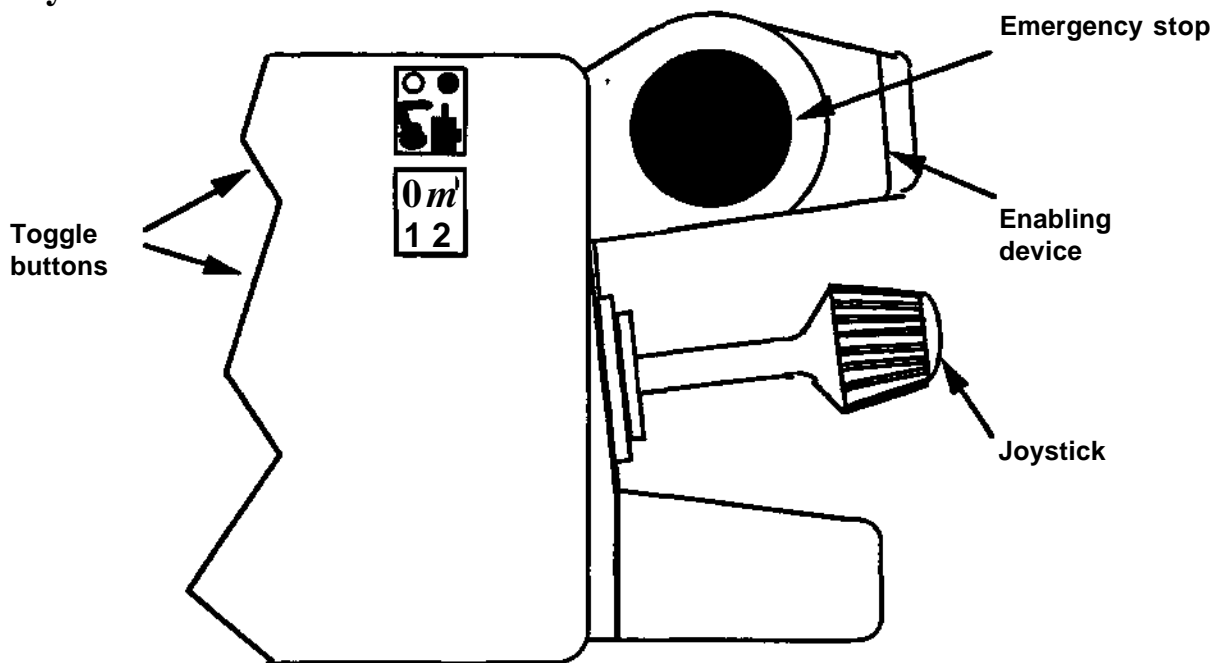
The five buttons located immediately below the display are designated multi-function buttons as their function depends on the current menu alternatives, displayed above. SCAN is used to page new alternatives forward on the display. SCAN is always at the extreme right in the menus.

When an alternative has been selected either a sub-menu or a question with response alternatives is presented. If a value is requested this is supplied by means of the numerical buttons. Complete program instructions are presented on the upper line of the display when the dialogue is finished.

An incorrect choice of main menu or menu alternative can most often be corrected by a reselection of menu or alternative. If the selection has resulted in the creation of a complete program instruction, this must be erased before any further work is performed.

10 different dialogue languages can be selected.

2.5 Joystick



The joystick on the programming unit is used for positioning of the robot, when programming. The joystick can also be used to change the positions of external axes which may be connected to the system. The joystick can also be used for override of the current and voltage when arc welding.

The joystick can be moved forward, backward, up and down and turned clockwise and counterclockwise or combinations of these. The speed of the corresponding robot movement is proportional to the deflection of the joystick.



The robot moves immediately when the joystick is operated. Make sure nobody is inside the robot working area before running the robot.

Two toggle buttons are associated with the joystick.

Either control of robot axes or control of external axes is selected by pressing the upper button. The functions of the lower button depend on the status of the upper button as follows:

- * If control of robot axes has been selected, selection of function 1 results in linear movement of the TCP if rectangular coordinates are active, or movement of axes 1-3, if robot coordinates are active. If function 2 is selected, wrist orientation can be performed around TCP, if rectangular coordinates are active, or, if robot coordinates, axis 4-6 can be moved.

- If control of external axes has been selected with the upper button, axes 7, 8 and 9 can be controlled with function 1. When function 2 is selected, the axes 10, 11 and 12 can be controlled.

Each time a toggle button is pressed, the selection changes. The LEDs in the buttons illuminate to indicate activation of the corresponding selection.

2.6 Enabling device

The enabling device is in operation when the operation mode selector is in either the MANUAL REDUCED SPEED or the MANUAL FULL SPEED mode.

When the enabling device is kept in the middle position, the robot is brought to the MOTOR ON mode.

When released or fully depressed, the robot returns to the MOTOR OFF mode.

2.7 Monitor (option)

A monitor, is used for display of:

- program section in the active program when program execution is stopped. The current instruction is pointed out by the cursor
- any comments in the program during program execution.
- error messages.

During program execution, the most recently executed comment will be presented on the screen until the next comment in the program is executed. (This assumes the presence of comments in the program.)

An example of a program section with comments is given below:

```
120 POSV=100%
130 ("WAITING FOR OBJECT*)
140 WAIT UNTIL INPUT 6=1
150 POS V = 100 %
160 ("COMPONENT 1 MOUNTED*)
```

2.8 Program printout (option)

This option makes it possible to connect a printer for printout of:

- Individual programs.
- The contents of the error buffer.
- Parameters

The communication with the printer is serial asynchronous at a speed of 300/1200 baud. The interface is RS 232C/V24. The printer connected must have an internal memory buffer.

2.9

Computer link (option)

Computer link makes it possible to connect a superior computer (S.C.).

The communication link, is serial asynchronous at 9600 baud. The interface is RS 232 C/V.24.

The S.C. can be of any type provided that it is equipped with the same interface and contains software to handle the same communication protocol as the robot.

Computer link with all of its functions provides added possibilities for flexible automatic production.

Simplified coordination and production supervision of complex processes is obtained with the help of functions for program transmission, remote control and status supervision.

Detailed documentation with respect to communication and applications functions is included in the document Computer Link.

2.10

Remote control

Remote control makes it possible to operate the robot from an external panel or from a PLC.

Digital I/O signals are used with a parallel or serial interface. The signals accessible are described further in chapter 4.12 and 4.13.

Section	Page
3.1 The Manipulator	3:3
3.2 Coordinate Systems	3:5
3.2.1 Coordinate systems for defining positions	
3.2.2 Coordinate systems in program execution	
3.2.3 Coordinate systems for manual movements	
3.2.4 Moving the robot with the joystick	
3.2.4.1 Rectangular base coordinate system	
3.2.4.2 Robot main axes coordinate system	
3.2.4.3 Robot wrist axes coordinate system	
3.2.4.4 Tool coordinate system, no basepoint active	
3.2.4.5 Tool coordinate system, basepoint active	
3.2.4.6 Reorientation around TCP	
3.3 TCP	3:18
3.3.1 Normal TCP	
3.3.1.1 Base point	
3.3.2 Room fixed TCP	
3.3.2.1 General	
3.3.2.2 Moving the robot with the joystick	
3.3.2.3 Program execution with fixed TCP	
3.3.2.4 Limitations	
3.4 Position programming	3:22
3.4.1 Robot configuration	
3.5 Movement velocity	3:23
3.6 Movement characteristics	3:25
3.6.1 Path optimization	
3.6.1.1 Path following principles	
3.6.1.2 Zones	
3.6.1.3 Velocity in corners	
3.6.1.4 Corner deviation	
3.6.1.5 Exceptions	
3.6.2 Speed optimization	
3.6.2.1 Guide text	
3.6.2.2 Programming	
3.7 Circles	3:36
3.7.1 Orientation interpolation	
3.7.2 Velocity in circle	
3.8 Soft Servo	3:38
3.8.1 Introduction	
3.8.2 Definition of softness	
3.8.3 Use of Soft Servo	
3.9 Program displacement	3:40
3.9.1 Parallel displacement with reference point (REFP)	
3.9.2 Program displacement with reference frame (FRAME)	
3.10 Singular points	3:46

3 Movement principles

3 MOVEMENT PRINCIPLES

3.1

The Manipulator

The robot's movement pattern can be briefly described as follows (see also the figure below showing the IRB 2000).

Axis 1 (C) Turning of the complete mechanical robot arm system.

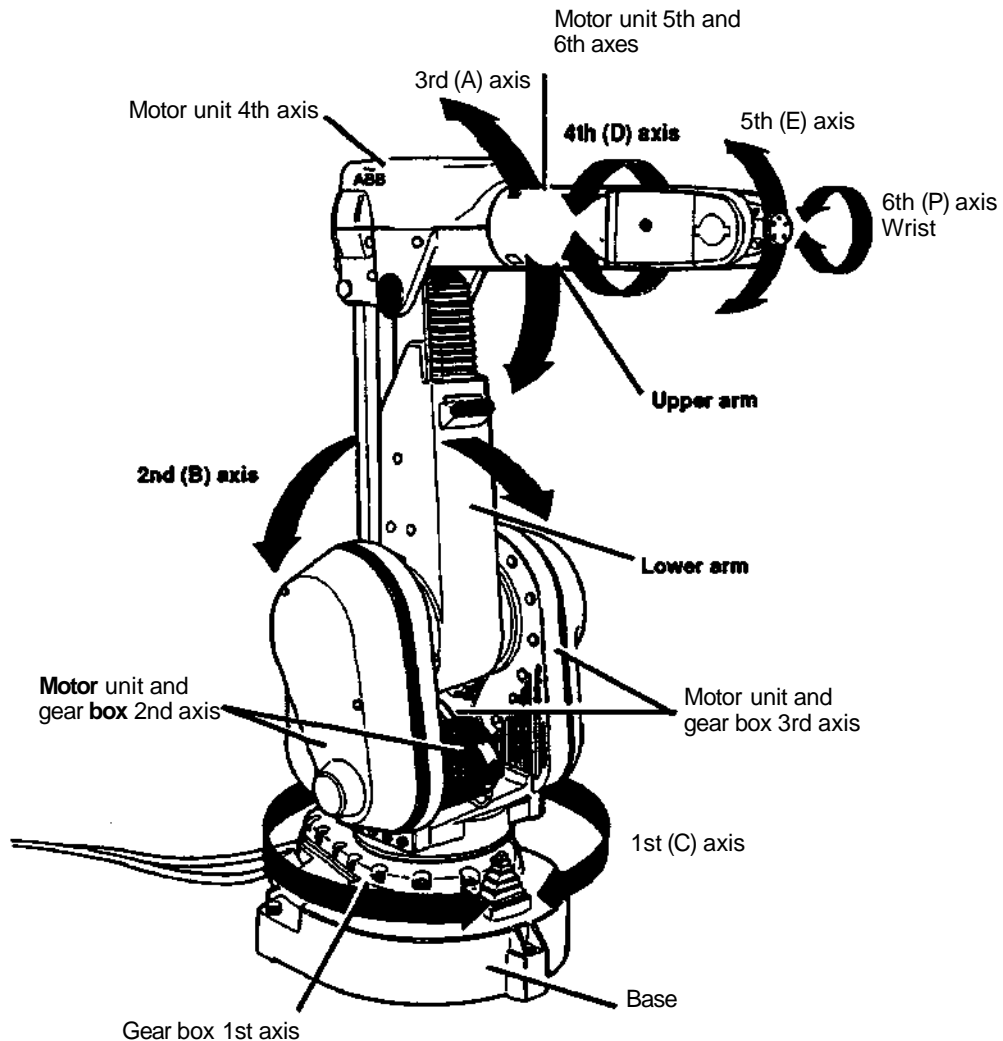
Axis 2 (B) Forward and reverse movement of the lower arm.

Axis 3 (A) Up and down movement of the upper arm.

Axis 4 (D) Turning of the complete wrist centre.

Axis 5 (E) Bending of wrist around the wrist centre.

Axis 6 (P) Turning of mounting flange.



Drive system

All motors are servo controlled, brushless AC motors, specially adapted for each axis.

Feedback system:

- one resolver per axis for position and speed indication
- a circuit board reading the resolver feedback from all axes

Brakes

The robot is supplied with brakes on all axes. The robot is automatically braked at emergency stops, power failure, or when taken to MOTORS OFF mode. Brakes are also activated after still-stand for 30 seconds (automatic operation) or 5 min (manual operation).



3.2 Coordinate systems

3.2.1 Coordinate systems for defining positions

All positions of the robot are expressed with the help of coordinate values. These values describe the positions of the robot in space. To know how to interpret the coordinates, i.e. to **know from** what reference and in what direction to measure the position, a reference frame or coordinate system must be defined.

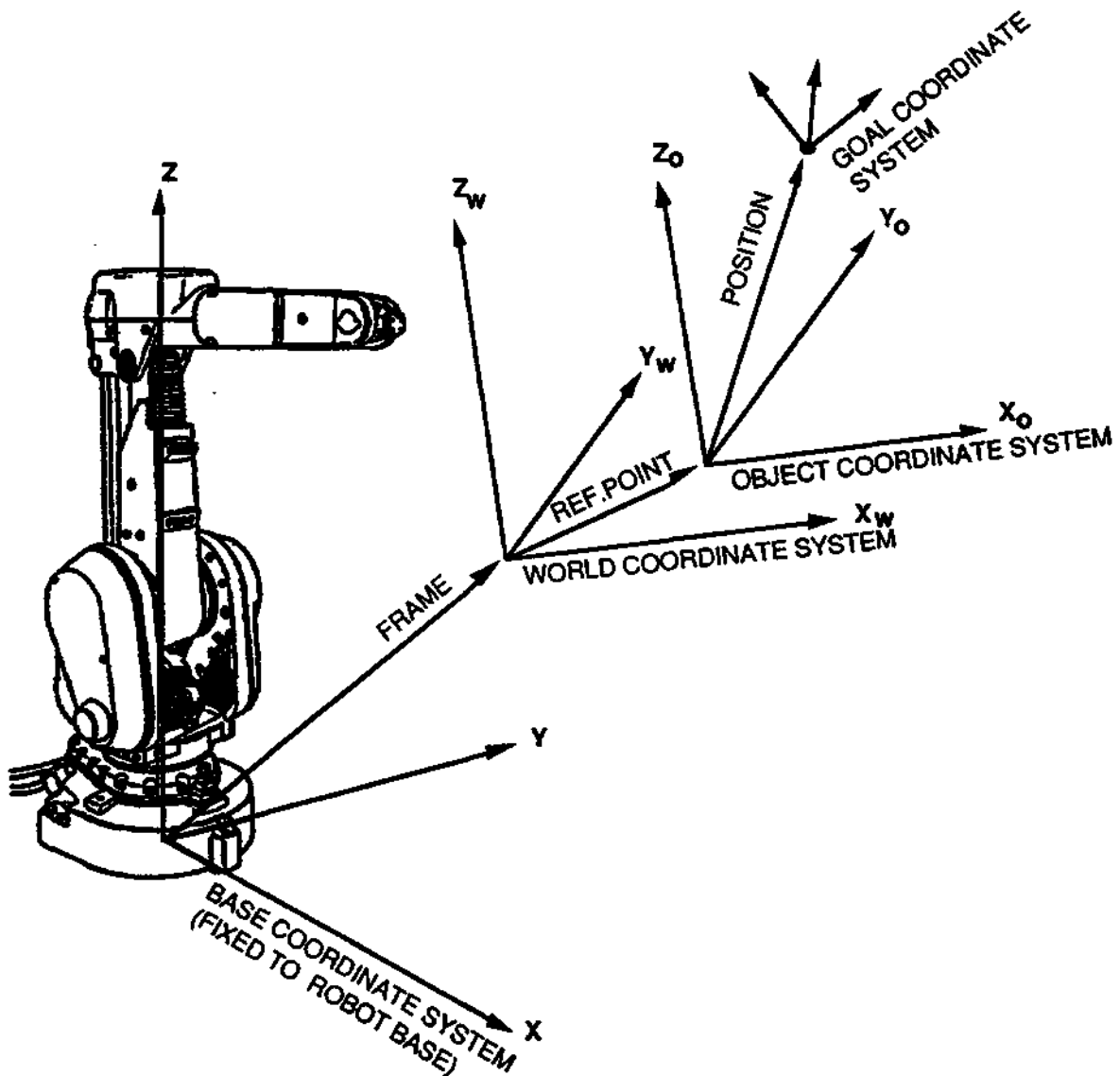


FIG.1 COORDINATE SYSTEMS USED TO DEFINE POSITIONS

3 Movement principles

The internal calculations in the robot controller will be expressed in the Base Coordinate System (BCS), see the figure above. BCS is always fixed to the robot base with the x-y plane in the floor and the z-axis pointing upwards and concentric with the first rotation axis.

However, it is not always a good idea to express the programmed positions in the BCS, because then it will be hard to change the positions if the robot installation changes. Often, when an installation changes, the relative movements are the same but the positions should be somewhat displaced. Then it is not necessary to reprogram all the positions but just to redefine the used coordinate system. For this purpose, a World Coordinate System (WCS) is used for defining the stored positions, see the figure above. The dislocation of the WCS from the BCS is defined with a FRAME-value, holding an x, y, z value defining the displacement, and an orientation quaternion (a four value vector) defining the reorientation.

In some cases a temporary displacement search will be needed, e.g. if an object is located at different positions from time to time and localized with a search function. In this case the position is expressed in the Object Coordinate System (OCS) and the displacement is defined with a REFERENCE POINT. Please note that REF. POINT displacement will only give a parallel displacement, i.e. no reorientation.

The outermost coordinate system of the figure above is the Goal Coordinate System (GCS). This defines the goal position (with an x, y, z value for the origin) and orientation (defined by a quaternion) which the robot should reach with its tool. The GCS is stored in every position instruction.

To summarize, the goal position and orientation is always stored in a position instruction relative to the OCS. The location of OCS is defined relative to the WCS, with REF. POINT. If REF. POINT is not active OCS and WCS will coincide. The location of WCS is defined relative to BCS with FRAME. If no FRAME is active WCS and BCS will coincide.

3.2.2

Coordinate system in program execution

While the final position or goal of a movement always is expressed in rectangular coordinates as have been described above, the path between positions can be executed in three different coordinate systems. These are Rectangular Coordinates, Robot Coordinates and Modified Rectangular Coordinates which can all be activated through instructions in a robot program. Each of these coordinate systems will give a slightly different path and should be used depending on priority of speed, accuracy or orientation.

Rectangular Coordinates

These coordinates are default and should be used in most movements. The path, which the TCP follows, will be linear or circular depending on the used position instruction. The speed will be controlled to the programmed value, but the path accuracy will be given priority. The speed will therefore be reduced when necessary. The orientation will be changed regularly from start orientation to end orientation. If start and end orientations are the same, the orientation will be held constant. A reduction in TCP velocity may occur during passage near a singular point (see section 3.9 and below), or during wrist reorientation.

Robot Coordinates

When running the robot in Robot Coordinates (Joint Coordinates) all axes are moved with constant motor speed from start position to end position. All axes will reach their goal position at the same time. The path will be curved in space. Robot Coordinates are used when a fast motion is given higher priority than the linear path accuracy.

Modified Rectangular Coordinates (MODRECT)

All six axis robots will have so called "singular points" in their working area. A singular point is a point where two or more of the robot axes are parallel. Very big robot axes movements or movement stop (with error message) can arise in or near singular points when running the robot in rectangular coordinates. In such cases it is advisable to use Modified Rectangular Coordinates to avoid the singular point.

When running the robot in MODRECT the TCP is moved along a linear path, as in rectangular coordinates. The wrist axes (axes 4, 5 and 6) however, are moved with constant speed in **Joint coordinates** from the start orientation to the end orientation. This means that the orientation will not be held constant during the movement, but will change slightly along the path.

Note: When changing from robot to rectangular coordinates the robot movement makes a short stop, while coordinating the axes. This stop does not occur when changing from rectangular to robot coordinates.

3.2.3

Coordinate systems for manual movements

Three coordinate systems are used to define manual (joystick controlled) movements of the robot. These are Base Coordinate System, Tool Coordinate System and Robot Coordinate System (see figure 2).

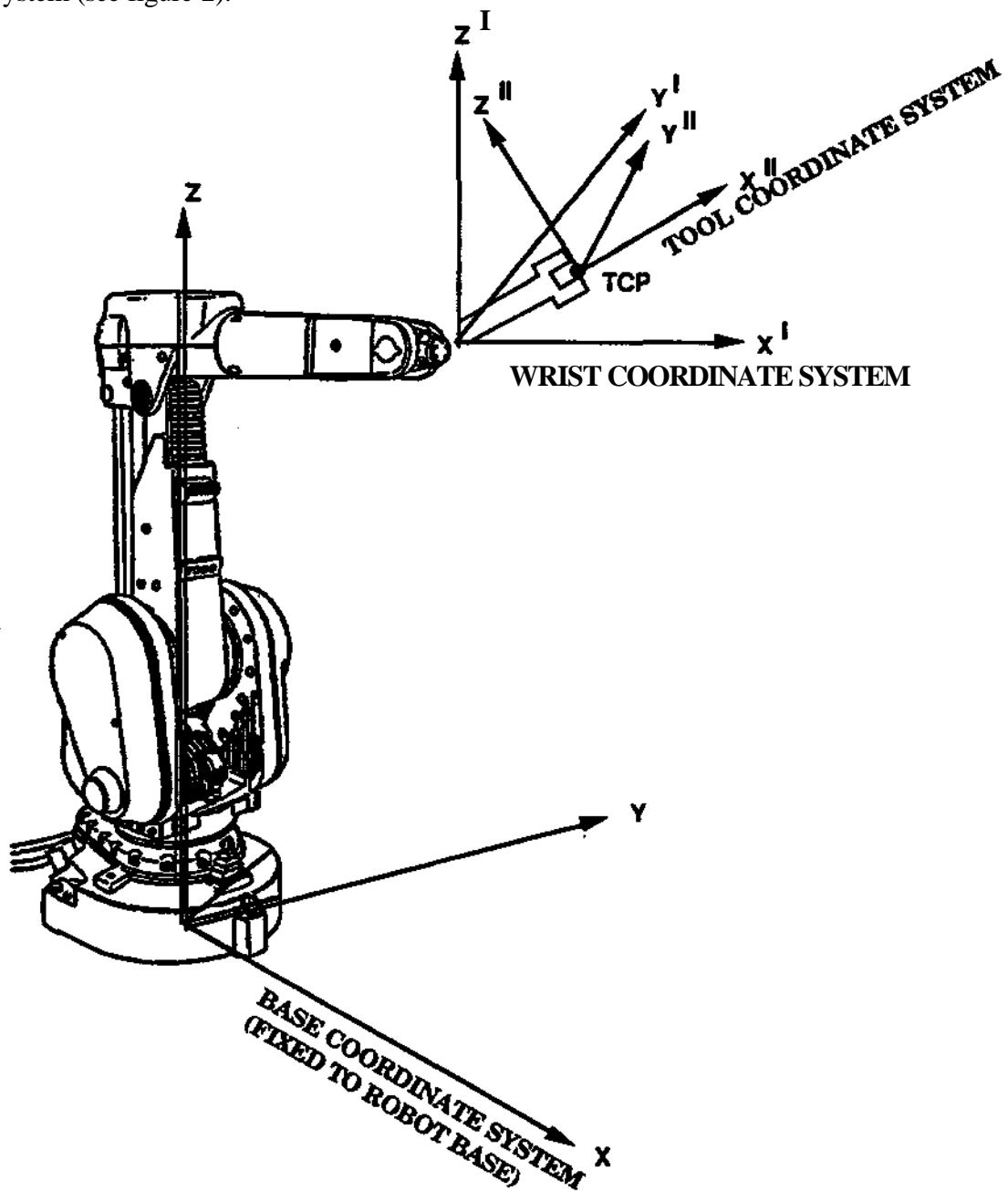


Fig-2 Coordinate system used for manual movements.

3 Movement principles

The Base Coordinate System is always fixed to the robot base with the x-y plane in the floor and the z-axis pointing upwards and concentric to the first axis. When using the Base Coordinate System, joystick movements will correspond to TCP movements parallel to Base axes.

The Wrist Coordinate System, (see figure 2) is used to define the tool and the Tool Coordinate System. The Hand Coordinate System is always fixed to the mounting flange of the robot, with the origin in the center of the flange, the x-axis pointing outwards and the z-axis pointing through the pinhole.

The Tool Coordinate System has its origin in the Tool Center Point (TCP) and is defined relative to the Wrist Coordinate System of the robot. The Tool Coordinate System is used to define the goal positions of the robot, i.e. the Tool Coordinate System is moved to coincide with the Goal Coordinate System (see 3.2.1).

Normally the orientation of the Tool Coordinate System is the same as for the Wrist Coordinate System. In this case only the TCP-values (x,y,z), defining the origin, are defined (see figure 3).

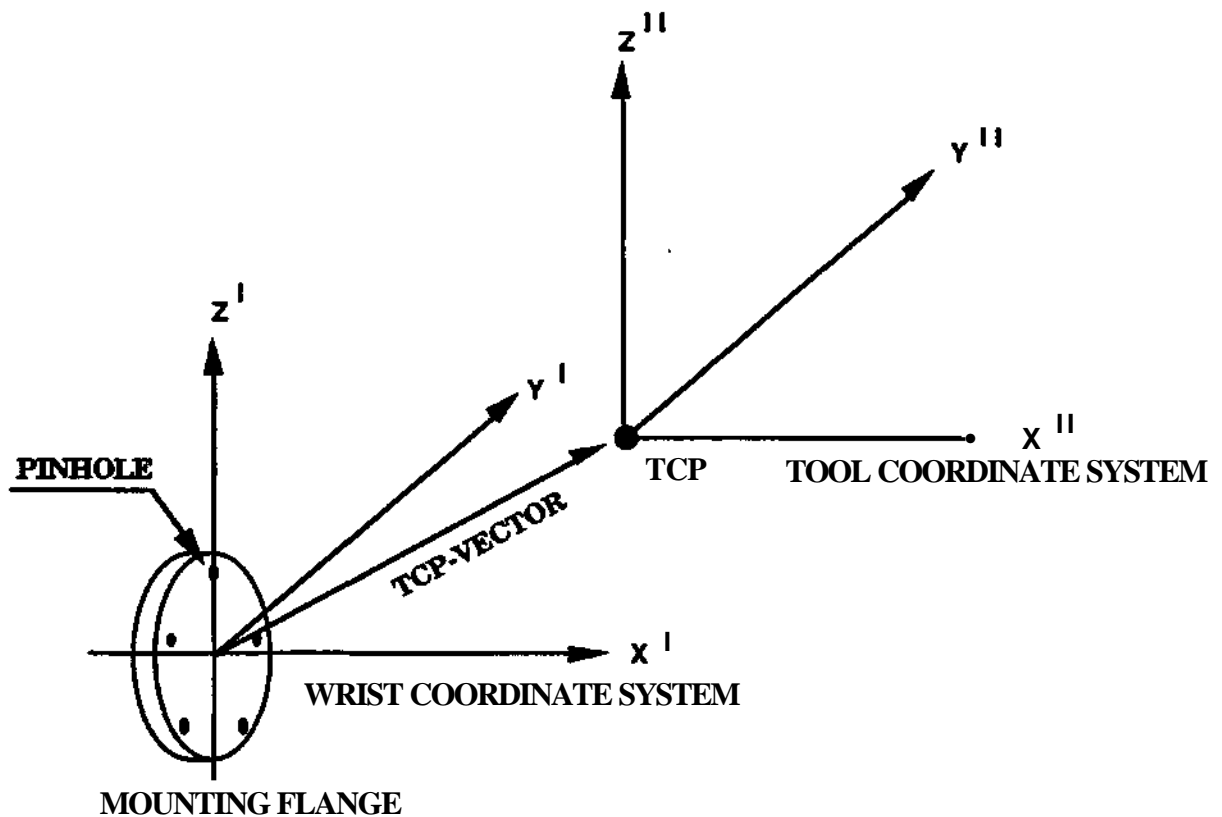


Fig. 3 Tool coordinate system with no base point

3 Movement principles

The Tool Coordinate System can, however, also be given another orientation according to the mounted tool. In this case also a Base point has to be defined (see figure 4). The orientation of the Tool Coordinate System will be determined in the following way:

- The x'' -axis will point through the TCP and the Basepoint.
- The y'' -axis is perpendicular to the x'' -axis and points through the TCP and on a line through the Basepoint, which is parallel to the x' -axis of the Wrist Coordinate System. If the x'' -axis is perpendicular to the x' -axis of the hand coordinate system, the y'' -axis will point in the negative x' -direction.
- The z'' -axis will be perpendicular to the x'' -axis and the y'' -axis.

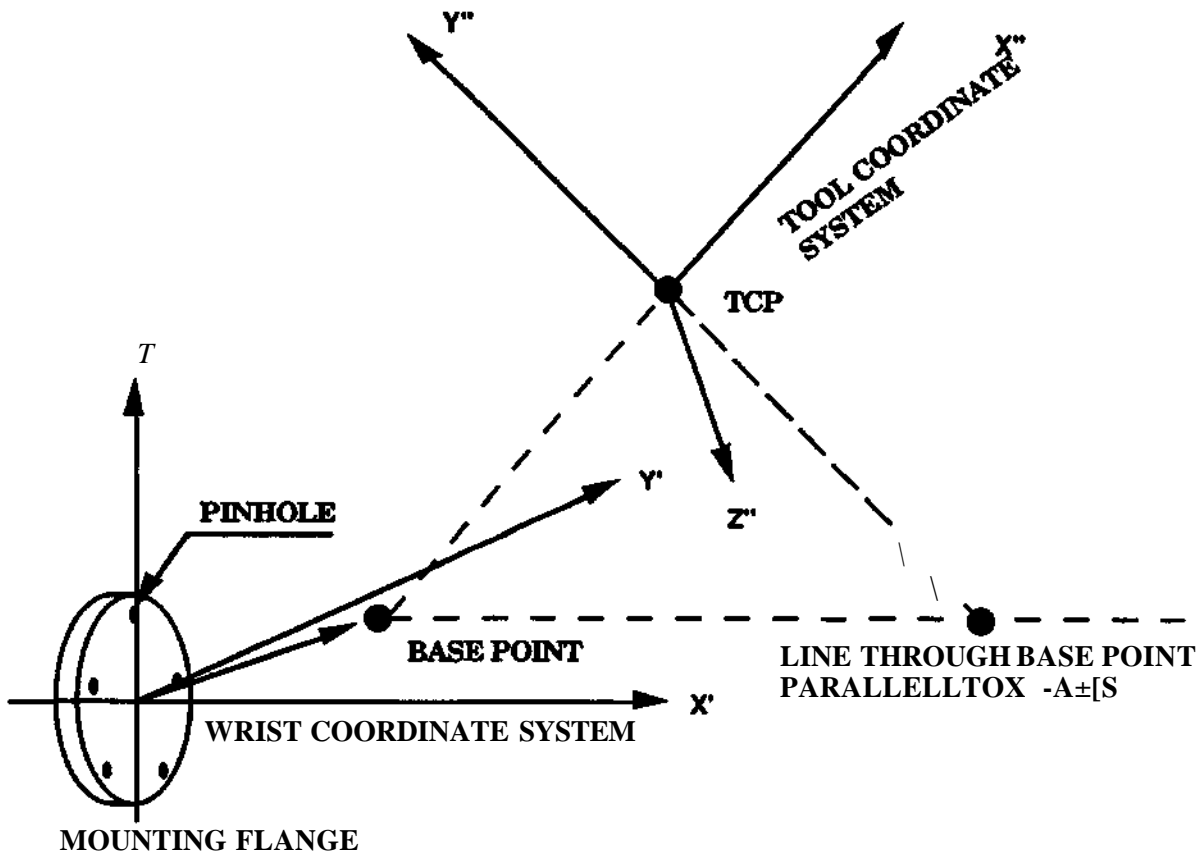


Fig 4 Tool coordinate system with base point

When using Tool Coordinate System, joystick movements will correspond to TCP movements parallel to Tool axes. In both Base Coordinates and Tool Coordinates, reorientation around TCP will be performed so that joystick movements will correspond to rotations around the Tool axes.

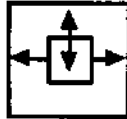
Robot Coordinates are used to control each individual axis of the robot. Joystick movements will correspond to movements of particular axis, as will be described later. Note that the use of different coordinate systems when manually moving the robot, will in no way affect the behaviour of the robot in automatic mode.

3.2.4

Moving the robot with the joystick

Select the coordinate system which is most convenient to run the robot to the required position with **the** joystick. (When programming positions, how the robot has reached a position is immaterial, it is the position itself that is registered).

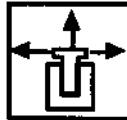
Rectangular base coordinates



Robot coordinates



Tool coordinates



The robot system has to be synchronized before coordinate system can be selected. How the movements of the joystick affect the movements of the robot in the different coordinate systems is described in the following sections.

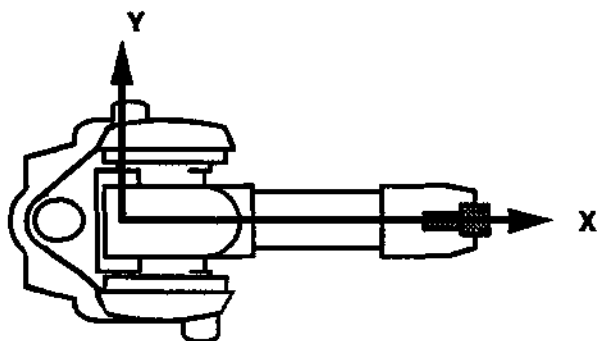
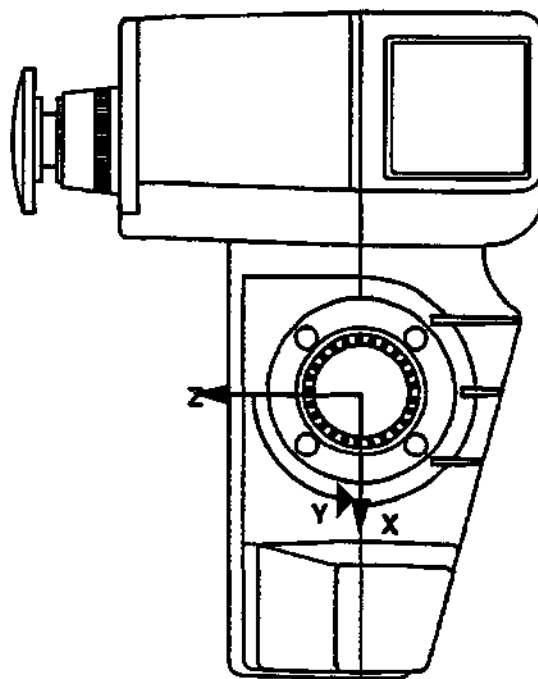
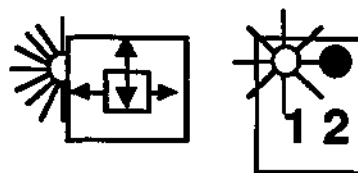
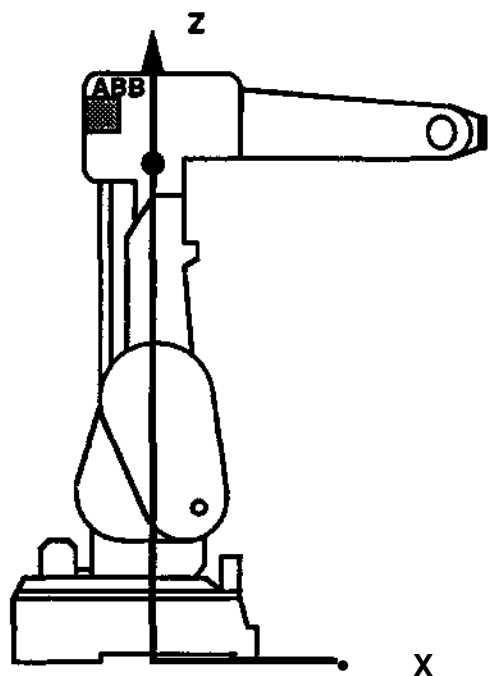


The robot moves immediately when the joystick is moved. Make sure nobody remains inside the robot working area before running the robot.

3.2.4.1

Rectangular base coordinate system

Obtained with button combination



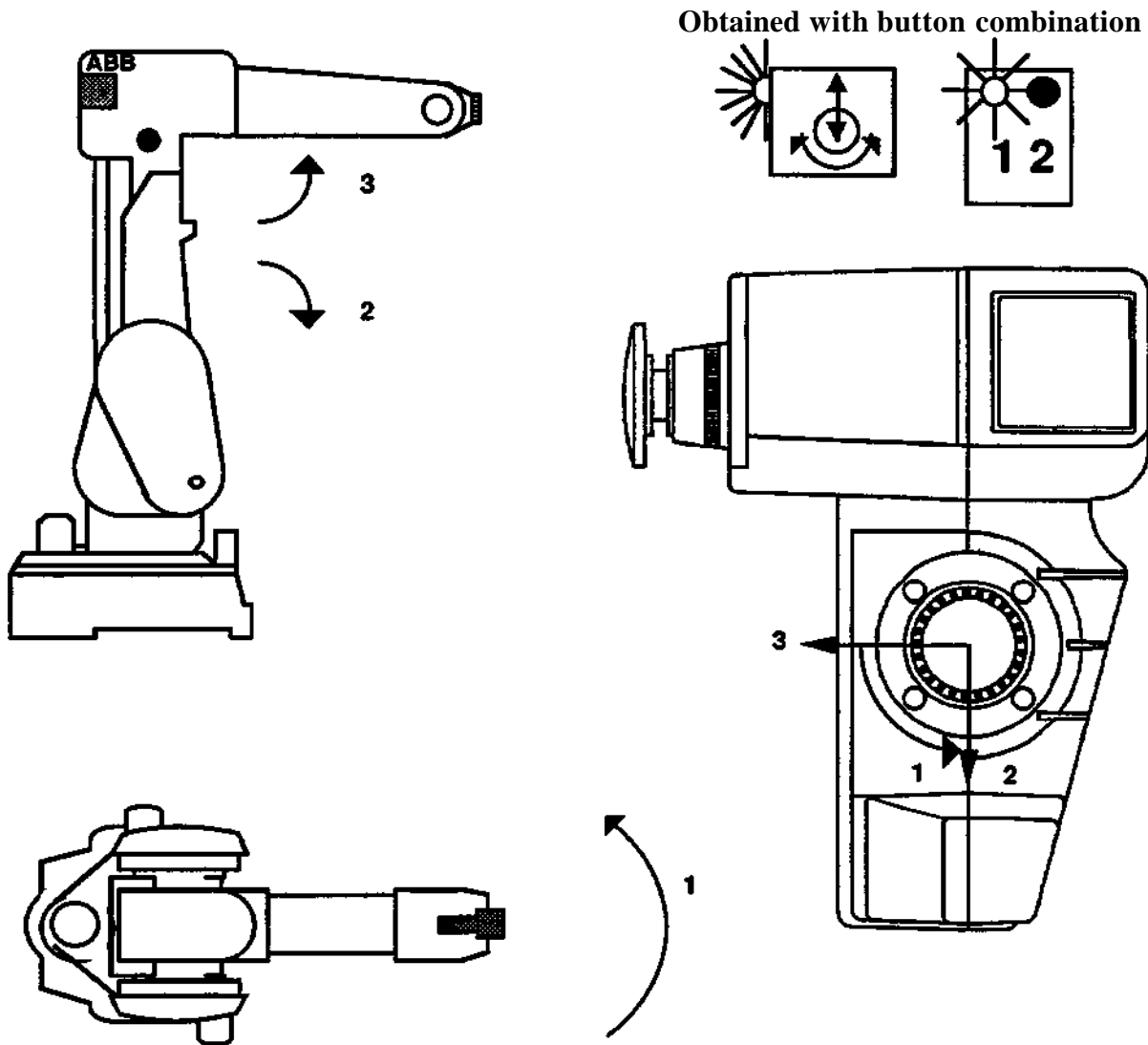
The coordinate system is permanently located at the origin in the robot base in accordance with the figures above. When the joystick is moved in the directions of the arrows, the robot TCP currently active moves in straight lines in the X, Y-and Z-directions respectively.

Note!

When the robot is mounted in an inverted position (function parameter MOUNTING=2), the movement directions of y and z are reversed relative to the robot base.

3.2.4.2

Robot main axes coordinatesystem



When the robot coordinate system is selected, joystick deflections in accordance with the arrows **correspond** to movements of the robot axes 1, 2 and 3 respectively in accordance with the **robot sketches**. The TCP does **not** move in straight lines.

The robot reacts to joystick deflection as described above also when the robot is not synchronized.



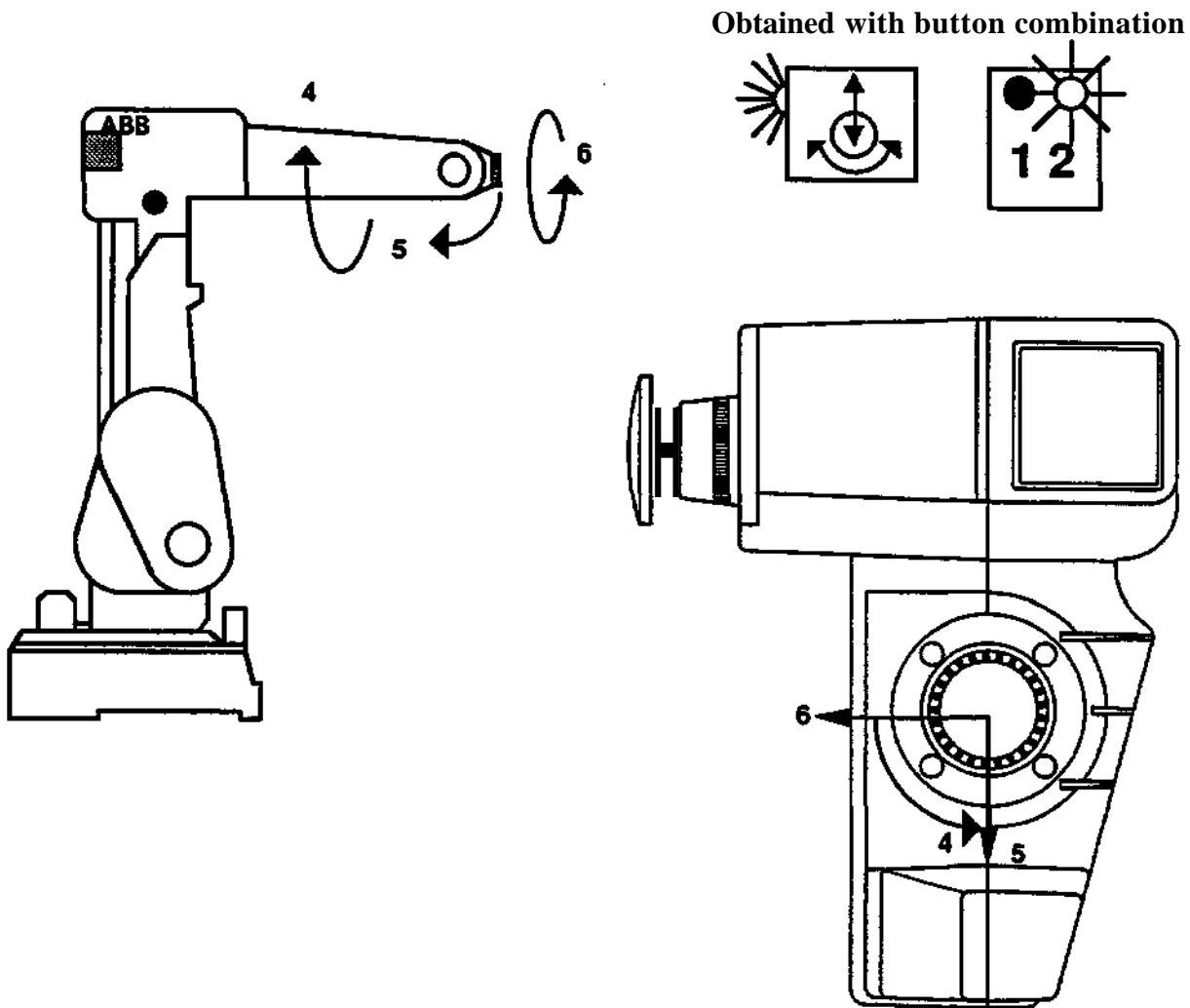
The working range of the axes is not supervised when the robot is manoeuvred unsynchronized. The movements are only interrupted by the mechanical stop if the robot attempts to move outside the working range.

Note!

When the robot is mounted in an inverted position (function parameter MOUNTING=2), the movement directions of axes 1 and 3 are reversed relative to the robot base.

3.2.4.3

Robot wrist axes coordinatesystem



When the robot coordinate system is selected, joystick deflections in accordance with the arrows correspond to movements of the robot axes 4, 5 and 6 respectively in accordance with the robot sketch. The TCP does not move in straight lines.

An unsynchronized robot also reacts to joystick deflections as described above.



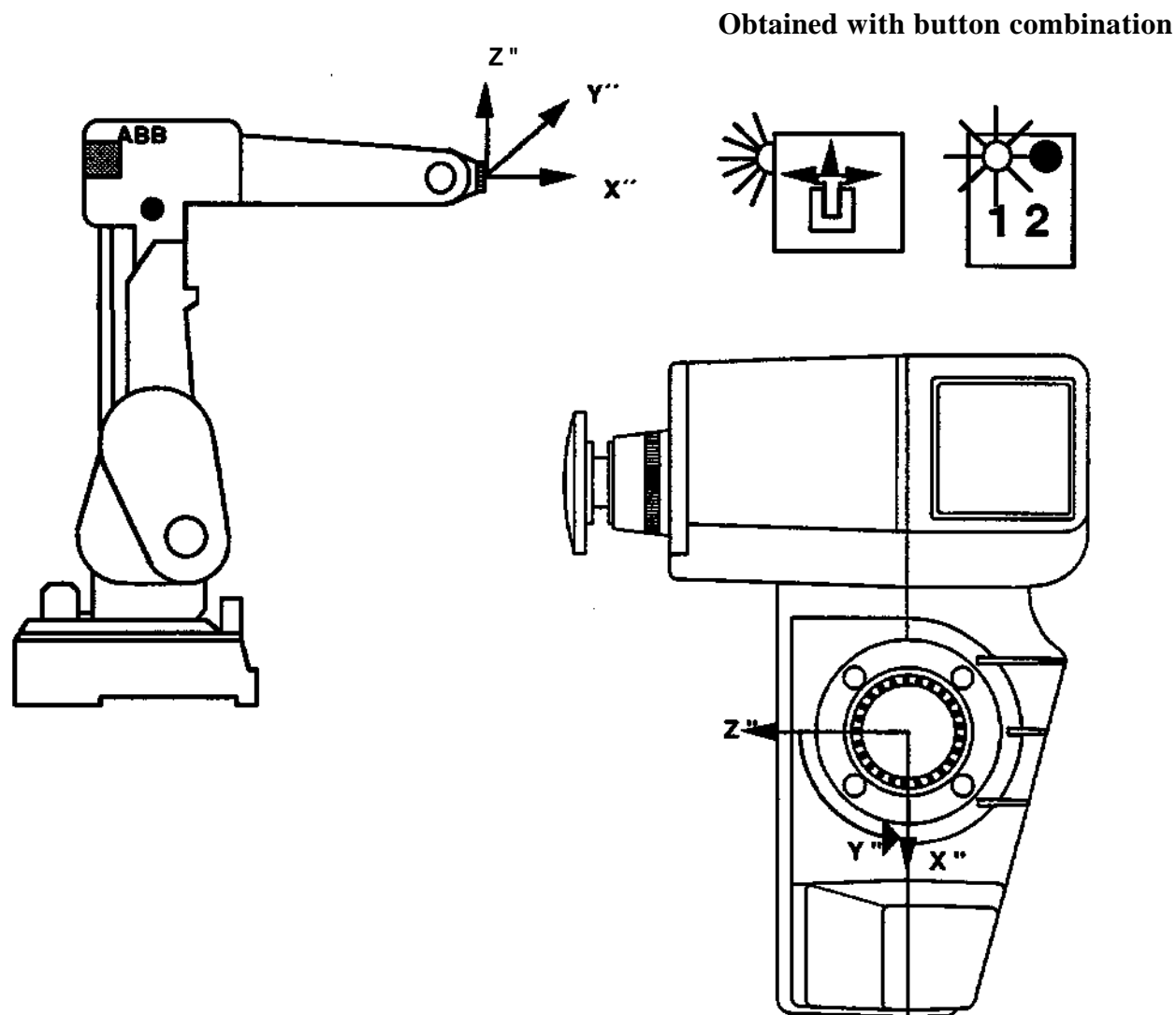
The working range of the robot axes is not supervised when an unsynchronized robot is maneuvered. Attempted movements outside the working range are only interrupted by the mechanical stop.

Note!

When the robot is mounted in an inverted position (function parameter MOUNTING=2), the movement direction of axis 5 is reversed.

3.2.4.4

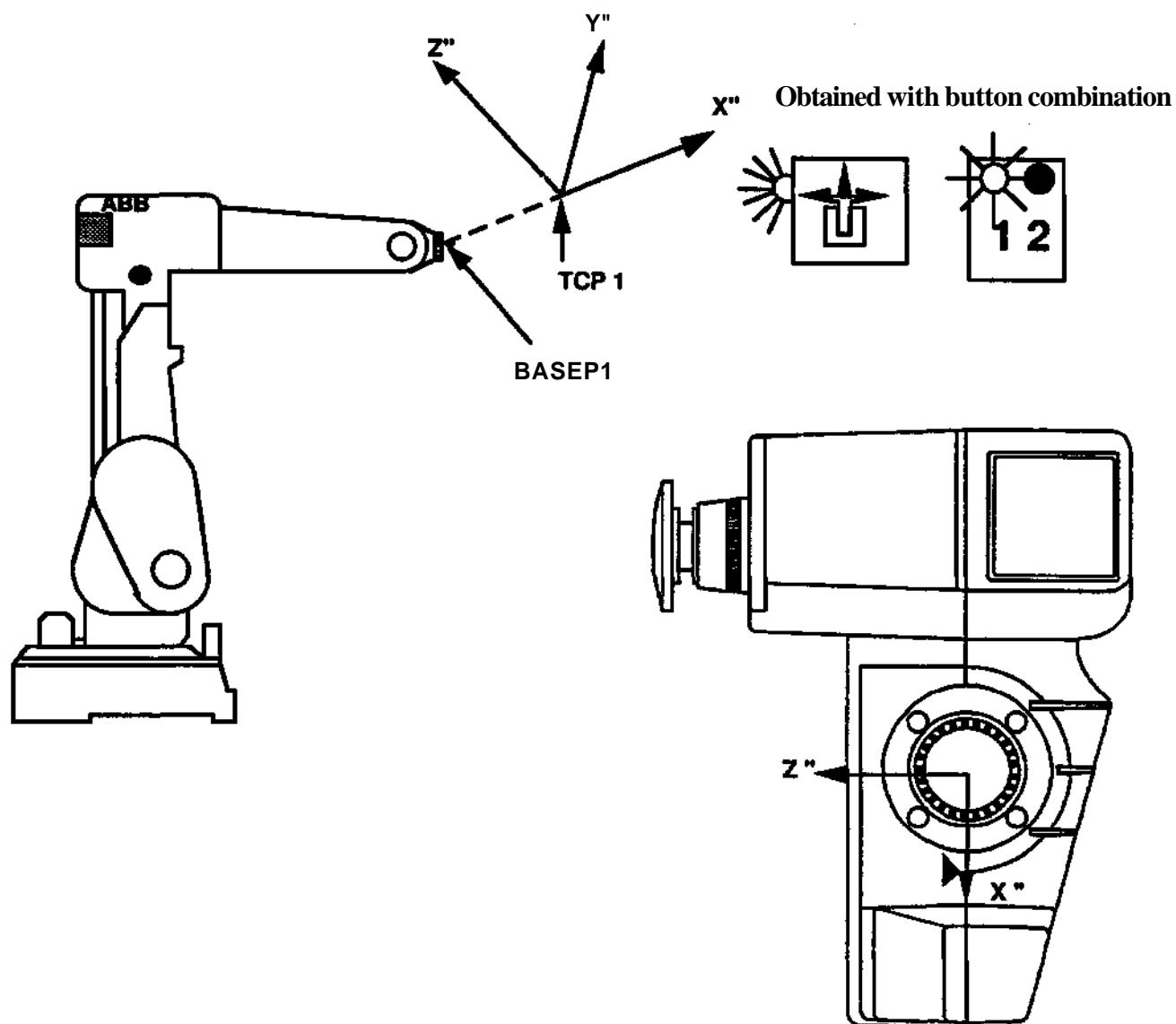
Tool coordinate system, no basepoint active



The tool coordinate system gives movements of the active TCP in straight lines, parallel to the axes of the tool coordinate system. If no basepoint is active the movement will be parallel to the wrist coordinate system, as shown above.

3.2.4.5

Tool coordinate system, basepoint active

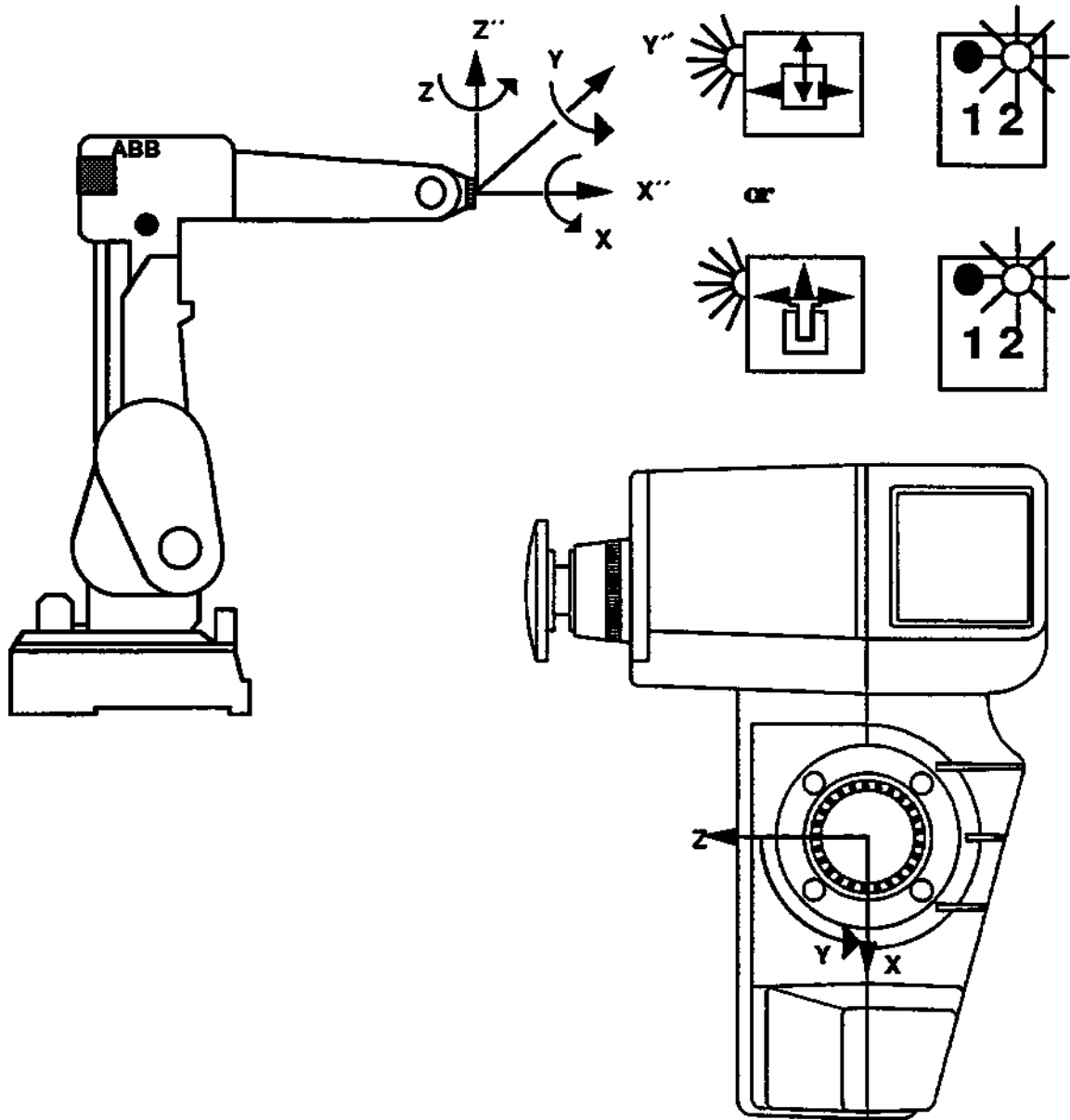


The tool coordinate system gives movements of the active TCP in straight lines, **parallel** to the axes of the tool coordinate system.

3.2.4.6

Reorientation around TCP

Obtained with button combination



The joystick deflections will rotate the tool around the axes of the tool coordinate system as shown above. The robot axes interact to keep the TCP stationary, despite the movements of the robot.

3.3 TCP

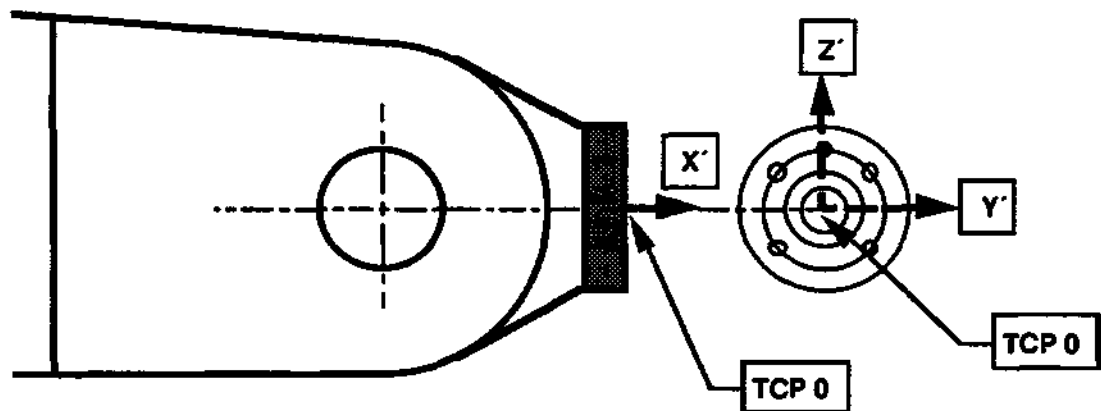
TCP is an abbreviation of Tool Center Point. Normally the TCP refers to the point moved by the robot (see 3.3.1). It can also refer to a room fixed point, around which the robot moves an object (see 3.3.2).

3.3.1 Normal TCP

The active TCP is the point moved by the robot when positioning and it is the position of the TCP which is registered when programming positions.

As many different types of tool with different forms and sizes can be mounted on the robot, the work point for each tool must be determined individually e.g. the nozzle of a glue dispenser, the centre point of a gripper, the tip of a drill or a deburring tool etc.

19 different normal TCPs can be stored simultaneously in addition to TCP 0 which is predefined as the centre of the mounting flange of the robot.



Only one TCP can be active at a time. TCP 0 becomes active immediately on start up and remains active until a change is made.

A TCP can be defined either manually or automatically. With manual definition, the operator states the measures in x-, y-, and z- directions from the center of the tool fixing (TCP 0) to the working point of the tool via the programming unit.

An example of automatic definition is the utilization of a fixed point in space to which first the TCP 0 and then the TCP to be defined are positioned. The robot system then calculates the difference between the TCP 0 and the new TCP, thus defining the new TCP.

When a TCP is redefined the original TCP-values are erased and the new measures are entered instead.

3.3.1.1 Base point

The BASE POINT is used in combination with TCP to define the "Tool Coordinate System", see section 3.2.3. This coordinate system is used to define not only the TCP of a tool but also its orientation, which is necessary if tool relative motions should be programmed or performed.

3.3.2

Room Fixed TCP

3.3.2.1

General

This section describes the function room fixed TCP. The function facilitates jogging of the robot, when the work piece is held by the robot and the tool is fixed. When reorienting the work piece, the movement is carried out in relation to the room fixed tool point (TCP), independent of the position of the robot.

When running the program, the work piece is going to move, either in a straight line or in a circle, when passing the room fixed tool point. The velocity of the work piece, relative to the room fixed point, is the programmed velocity.

10 different room fixed TCP's can be stored.

The function is not available in the arc weld software.

3.3.2.2

Moving the robot with the joystick

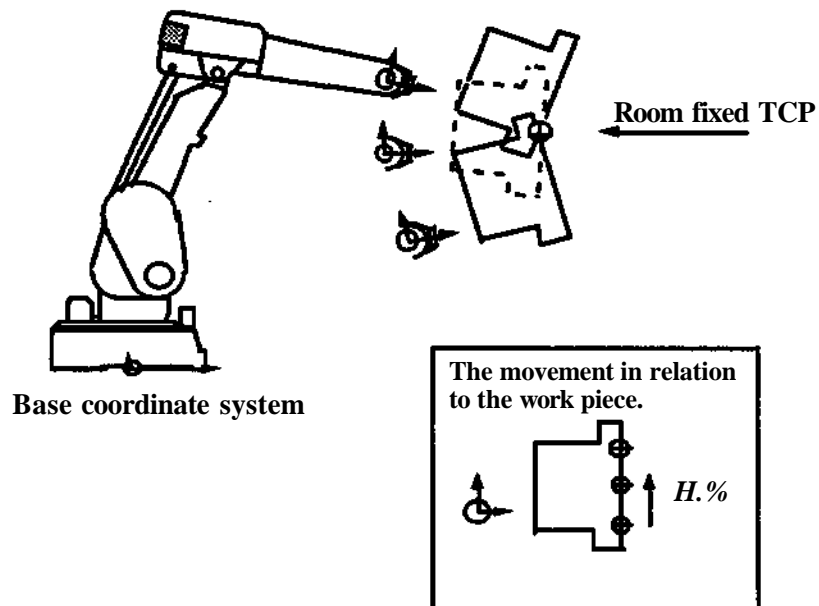
The robot will move normally in the base, wrist and robot coordinates, when jogging the robot. The difference, compared to a normal TCP, is when reorientation of the wrist is performed. Then the work piece will twist and move in relation to the room fixed work point, independent of the position of the robot. See figure next page.

3.3.2.3

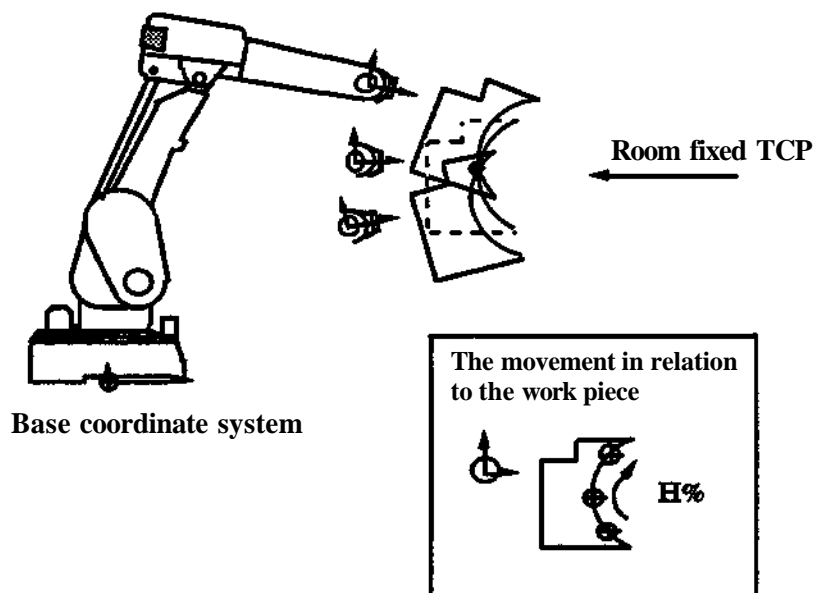
Program execution with room fixed TCP

When running the program the work piece is going to move either in a straight line or in a circle while passing the fixed tool point. The path and velocity are not affected by reorientation of the work piece during the movement. See figure below.

Linear interpolation with room fixed TCP.



Circular interpolation with roomfixed TCP



3.3.2.4 Limitations

BASE POINT

Definition of base points is not possible for a room fixed TCP. An error message is displayed if an attempt is made.

REFERENCE POINT

The reference point does not move the room fixed TCP. Instead the work point on the work piece is moved. The reference point must be deactivated before changing between an ordinary and a room fixed TCP.

FRAME

Manual and automatic definition of frame can not be done with an active room fixed TCP. An earlier defined frame can be used. The frame moves the room fixed TCP and changes the orientation of the work piece in the same way as for an ordinary position.

MIRROR

The function is interrupted with an error message if a position is programmed with a room fixed TCP.

WEAVING

Weaving is blocked for programming and program execution when a fixed TCP is active.

MODPOS

The function MODPOS is working normally when a room fixed TCP is active, but if the position instruction in question was created with an ordinary TCP, the function is blocked.

RELTOOL

The function is blocked for programming and program execution if a room fixed TCP is active.

VISION

If a room fixed TCP is active all the vision functions are blocked for programming and program running.

3.4

Position programming

The following principles apply to position programming.

- The robot, i.e. the TCP, is moved by deflecting the joystick, to the position it is to pass when the program is executed.

The position, i.e. TCP position and wrist/tool orientation (see below) expressed in the world coordinate system (see 3.2.1), is programmed and stored in memory by pressing the POS-button.

When a position is programmed:

- The wrist orientation is stored, if no basepoint is defined for the active TCP.
- The tool orientation is stored if a base point for the active TCP is defined.



It should be noted, that a position can be programmed with one TCP active and later a movement to that position could be performed with another TCP active. Generally this should be avoided, since it could result in dangerous and unexpected movements.

Sometimes this could be useful, however, giving a possibility to use different tools in the same position, which has to be programmed only once. In most cases the different tools have different mounting orientations relative to the mounting flange, and in these cases base points must be used to define the tool orientations.

When a position is programmed, a number of optional arguments describing the movement, may also be defined. Such arguments are for instance corner zones (see 3.6) and speed (see 3.5). For a complete description of the position instruction, see chapter 6.

3.4.1

Robot configuration

Many combinations of tool position and orientation can be reached with more than one robot configuration. The controller therefore has to make a choice of configuration, before going to the position in question.

Information on the configuration at programming time is stored in each position instruction.

Rectangular coordinates

The configuration in the next position that gives the smallest wrist axes movements is chosen.

The configuration is supervised so that execution stops and an error message is issued if either

- the configuration obtained as the next position is reached is not the same as the one at programming time if the position is a FINE-position.
- on the way towards the next position, the axis 4 deviates more than 45 degrees, or axis 6 deviates more than 90 degrees from the start position in direction away from the stored configuration of the next position.

This supervision can be deactivated (see 8.5) so as to allow the configuration to be different at programming and execution time.

Robot coordinates and modified rectangular coordinates

The configuration which is closest (normally identical) to that at programming time will be chosen.

Displacement

If some kind of position displacement is made (reference point or frame active, pallett instruction, stored position with offset or mirrored program), the configuration information in the position is no longer used (as it is not valid for the displaced position).

This means that, regardless of coordinate system, the configuration in the next position that gives the smallest wrist axes movements will be chosen. There will be no supervision of the configuration.

☐ **Note!** When mirroring a program the operator is given the option of retaining the configuration information.

☐ From software version M93/7 and on, the configuration handling for displaced programs (except for mirrored programs) can be changed by the parameter DHANDCHK. If set to 1 the following apply:

Displaced and undisplaced programs are handled in the same way. This change will lead to some positions causing the supervision to trip in rectangular coordinates. To avoid this, the supervisions can be deactivated for such positions (see section 8.5).

The advantages of the changed handling in displaced programs are:

1. The user can chose whether configuration supervision should be active or not (always deactivate previously).
2. The user can choose between the programmed and the closest configuration in modified rectangular coordinates and in robot coordinates (by chosing DHANDCHK active or not). Previously the closest configuration was always chosen.

☐ 3.5 Movement velocity

☐ The speed of robot movement during program execution depends on the basic and maximum speeds, programmed in a velocity instruction, and the speed percentage specified in each positioning instruction. The speed is also affected by the speed correction (the plus and minus buttons on the programming unit). The preprogrammed basic speed is 1000 mm/s and the preprogrammed maximum speed is 2500 mm/s.

The positioning speed is specified in each instruction as a percentage of the basic speed. There is however a check that this never exceeds the maximum speed. If an attempt is made to program an impossible maximum speed, an error message is obtained and the value is rejected.

Note that speed correction, with the plus (+%) and minus (-%) buttons, is only to be used for testing of different speeds. The percentage should always be set to 100% when the finished program is run in production. Otherwise unnecessary speed reduction and jerks can result in corners.

Rectangular coordinate system

The programmed velocity refers to the velocity of the TCP. The robot will always try to reach the programmed velocity, but due to physical limitations, this might not always be possible for high speeds.

If minimum cycle time is strived for (and robot coordinates cannot be used), it is advisable to program a realistic velocity. If a velocity which cannot be reached is programmed, unnecessary delays will be introduced in corners, which will lead to a longer cycle time than would have been the case with a lower programmed speed.

Wrist reorientation and external axes movements are synchronized with the TCP so that they are spread evenly over the available path length and all conclude the movement simultaneously. For large wrist reorientation, the reorientation can set a limit to the TCP velocity to enable the programmed orientation to be reached at the programmed position. At low percentage speeds (V% below 10%) the rate of reorientation is reduced proportionally to the percentage speed value. Note that the rate of reorientation is also affected by the \pm % buttons.

The external axes velocities are limited by the instruction velocity.
V=100% corresponds to maximum external axes velocity.

Influence of programming functions on the robot velocity:

	Programming functions		
	Programmed velocity V	Programmed POS-speed V%	Speed correction + %-%
TCP velocity	x	x	x
Reorientation velocity	-	below 10%	x
External axes velocity	-	x	x
Velocity dependent zone	x	x	-

Robot coordinate system

When robot coordinates are active, the robot moves at approximately the speed programmed.

To guarantee the shortest possible cycle time, the maximum (2500 mm/s) speed should be programmed. All axes concerned begin and conclude their movements simultaneously and they all run at constant speed.



The speed of external axes and the robot speed when running in robot coordinates may supersede the programmed speed!

3.6

Movement characteristics

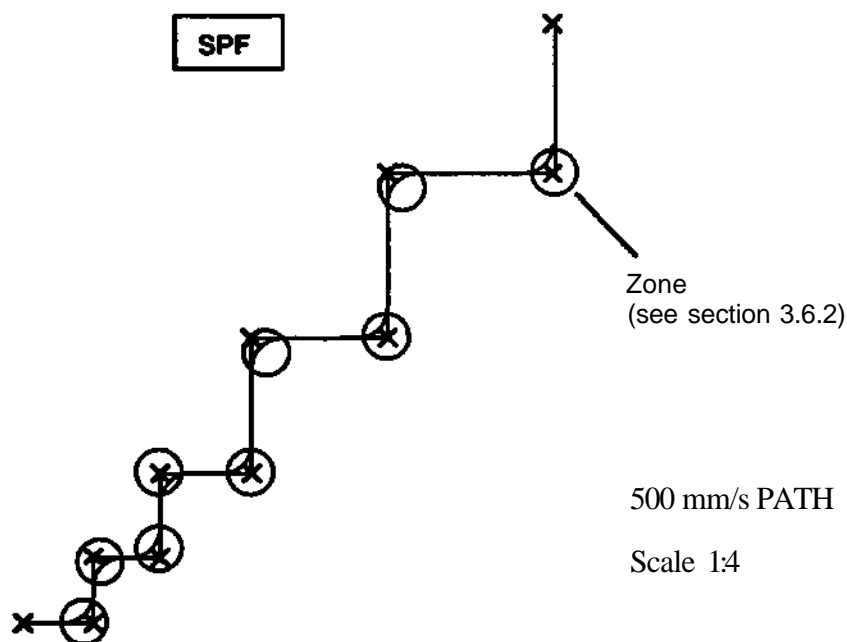
Two different modes of path planning can be chosen with the system parameter PATH (see "Installation S3"). These are path and speed optimization.

3.6.1

Path, optimization

To ensure accurate path performance with the high speed and agility of ABB robots, the S3 controller uses the so called SPF (servo path following) concept

SPF results in optimum path accuracy in rectangular and modified rectangular coordinates.



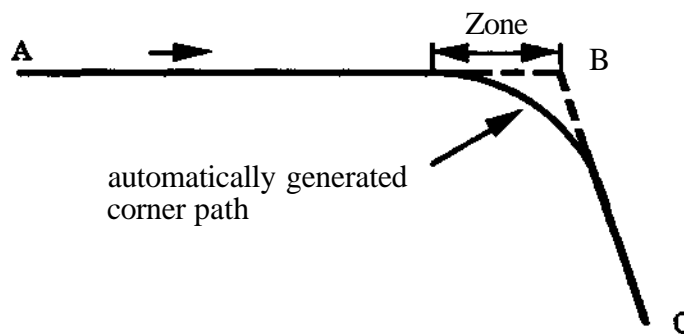
3.6.1.1

Path following principles

SPF keeps the TCP close to the straight line between two programmed positions while compensating for changing moment of inertia, gravity and robot dynamics without overshoot.

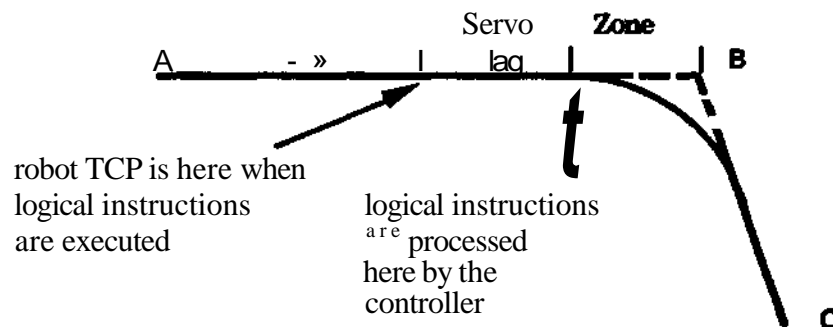
Path following in rectangular coordinates is carried out by continuously coordinating the movements of each individual axis to the speed of the axis which limits the performance for the moment. When a change of direction in TCP movement is programmed, a parabolic corner path is automatically generated.

The corner path starts at a defined distance from the programmed position. This distance is called the corner zone or just zone.



If the robot TCP velocity is too high to take the corner, the incoming velocity is reduced to a suitable value. A certain speed reduction will also take place within the corner zone.

Because of servo lag logical instructions are executed just before the robot TCP enters into the zone.



3.6.1.2 Zones

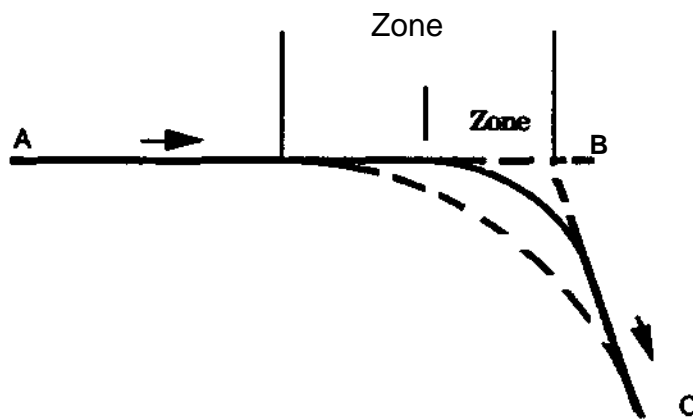
The zone is the distance from the programmed position to the point at which the robot TCP leaves the straight-line path. The different zone sizes are defined in the system parameters (S3 Installation) and a suitable zone is selected when programming a position (see 6.3).

The size of the zone is defined in mm or inch, however the value is applicable only when using rectangular coordinates. When using robot coordinates the zone size might differ somewhat from that programmed.

The zone can have a fixed size or a velocity dependent size.

The zone has fixed size if the zone size is entered as a positive value. If the value is entered as a negative value, the zone will be treated as a velocity dependent one.

A velocity dependent zone size corresponds to the defined value (in mm or inches) at a TCP velocity of 1000 mm/sec and is directly proportional to the velocity. For example in rectangular coordinates and defined zone value of -25 mm, the zone size at 100 mm/sec is 2,5 mm.



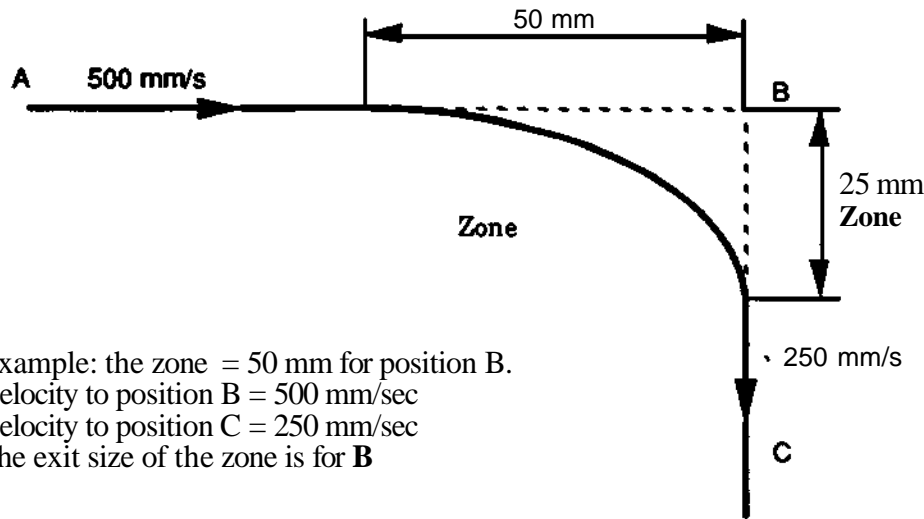
The figure shows zone size variation due to either:

- change in zone definition for position B
- change in programmed TCP velocity (if velocity dependent zone is used for B)

The size of a velocity dependent zone is proportional to the programmed velocity VELOC (section 5.5) and the programmed percentage velocity V % (section 6.1). It is not affected by the speed correction (-%) (+%) buttons on the programming unit thus, retaining an unchanged path when testing a program at different speeds.

3 Movement principles

If the velocities before and after a programmed position differ, the zone entry and exit sizes of a velocity dependent zone will differ, and will be directly proportional to the velocity ratios. The path is automatically generated accordingly.



Example: the zone = 50 mm for position B.
Velocity to position B = 500 mm/sec
Velocity to position C = 250 mm/sec
The exit size of the zone is for B

$$\frac{250}{500} \times 50 = 25 \text{ mm}$$

Using zones when programming positions

One of four different corner zones can be selected in a position instruction with the argument Fine, Path or Corner 1/2. The different zone types will be briefly described in the following:

1. Position Fine

With argument FINE, the robot TCP moves towards the programmed position, pauses until all axes have entered, and then goes on to the next instruction. Logical instructions are carried out within the defined zone.

2. Position Path, C1 or C2

With argument PATH, C1 or C2, a parabolic path is automatically generated at corners. The TCP moves through the corner without halting. Logical instructions are carried out before entering the zone as described in 3.6.1.1. With default settings PATH signifies a normal and C2 a big speed dependent zone whereas C1 signifies a small fixed size (see 6.3 for default values).

A corner can also be programmed with a CIRCLE instruction. In this case three positions have to be programmed to construct the corner. The result will be a circular arc instead of a parabola. The deviation from the corner will be bigger than when using zones. The speed through the corner will be constant in contrast to the zone case (see 3.6.1.3).

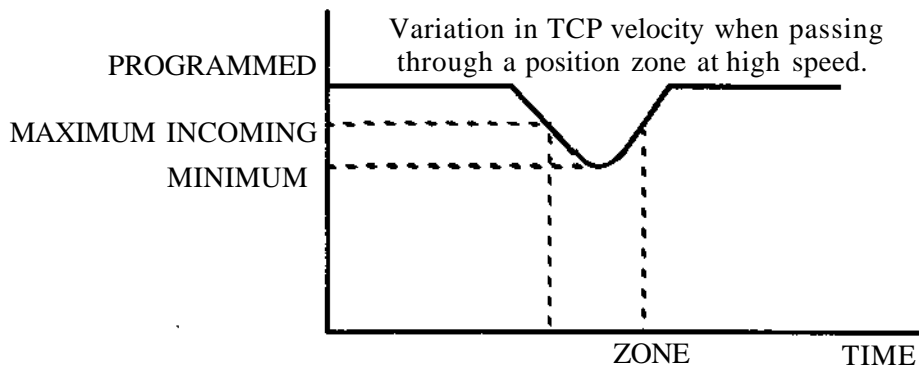
3.6.1.3

Velocity in corners

Highest incoming velocity

If necessary, the velocity when entering the zone is reduced automatically to the maximum velocity allowed to perform the programmed path without overshoot. The maximum velocity is a function of the zone size and the corner angle. It can also be limited by large wrist reorientation.

VELOCITIES



Approximate maximum incoming velocity mm/sec (without large wrist reorientation)

Zone size (mm)	Corner angle				
	0 °	45 °	90 °	135 °	180 °
1	40	40	20	15	15
2	80	80	40	30	30
5	200	200	100	70	70
15	600	600	300	200	200
25	1000	1000	500	350	350
50	2100	1000	700	500	400
100	2500	2100	800	700	600

Velocity through the curve

The optimum acceleration and deceleration performance of the axes is used through the corner. This results in a smooth reduction of velocity along the curved path to a minimum value, and an increase again to the exit value.

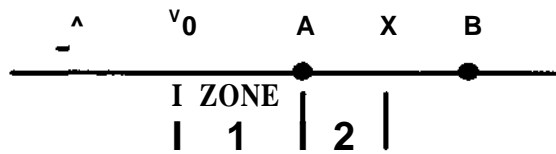
The minimum velocity is proportional to the incoming velocity and is a function of the corner angle. It can also be limited by large wrist reorientation.

Corner angle	minimum velocity in % of incoming velocity
0°	100%
45°	90%
90°	70%
135°	40%
180°	0%

3 Movement principles

Velocity between overlapping zones.

If two positions lie so close that their programmed zones overlap, there will be an automatic reduction of zone sizes, so as to avoid overlap. The TCP velocity between the two positions will then also be reduced. The reduction is directly proportional to the reduction of zone size, see the figure below.



Entry zone to position A = ZONE1

EXIT zone from position A = ZONE2

The $X = V_0 \times \frac{\text{ZONE2}}{\text{ZONE1}}$

Example: The original zone is 15 mm, the positions are 15 mm apart so that the exit zone size for position A is 7.5 mm. The velocity at X will then be $7.5/15 = 0.5$ times the programmed velocity.

The speed correction buttons -% and +%

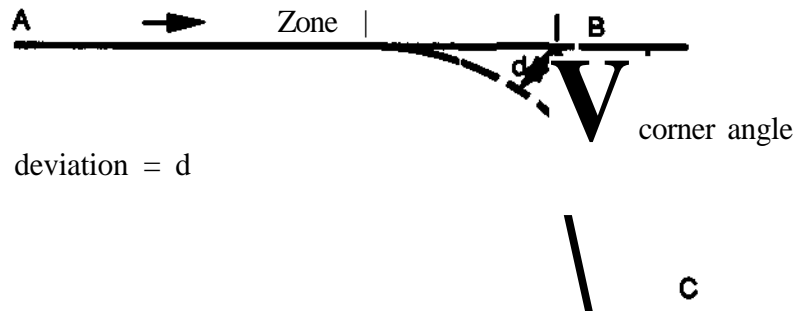
As mentioned earlier, the speed correction buttons do not affect the size of zones, and the TCP path is therefore unchanged.

However, the TCP velocity and the velocity of external axes are affected by these buttons, as well as the rate of reorientation. This will give the possibility to check the path with a lower speed. This possibility should only be used at testing. The speed correction should be set to 100% in production.

3.6.1.4

Corner deviation

The deviation from the corner point is a function of the zone size and the corner angle.



Deviation in nun

Zone size (nun)	Corner angle				
	0°	45°	90°	135°	180°
1	0	0,2	0,4	0,5	0,5
2	0	0,4	0,7	0,9	1
5	0	1	1,8	2^	2,5
15	0	3	5,5	7	7,5
25	0	5	9	11,5	12,5
50	0	10	18	23	25
100	0	19	36	46	50

Note that for velocity dependent zones, the current velocity must of course be taken into account to calculate the actual zone size.

3.6.1.5 Exceptions

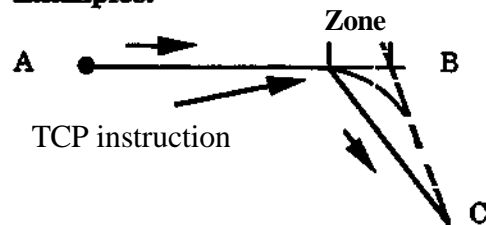
No corner path is generated after the following instructions and functions:

- after a TCP instruction (section 5.9)
- at a FINE position (section 6.3)
- after a SEARCH STOP (section 6.4)

between two circle segments (section 6.13)

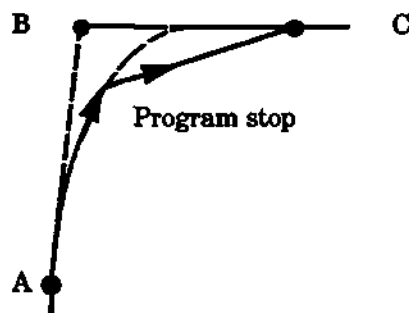
- on PROGRAM START after a PROGRAM STOP in a zone (section 7.1)
- on INSTRUCTION START (section 7.2)
- on BACKWARD instruction (section 7.3)
- on WEAVE (section 6.5, 12.4)
- on LASERTRAK operations (separate document)
- on manual control with the joystick (section 2.5)

Examples:



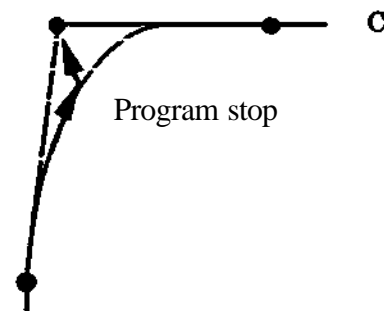
POS A FINE
POS B PATH
TCP
POS C FINE

The TCP is activated at the entry point to the zone and moves directly to the next position (argument FINE recommended)

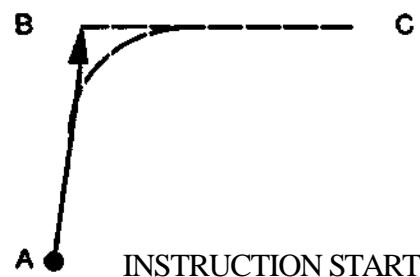


Program stop within the zone of position B.

Forward movement
to C after start



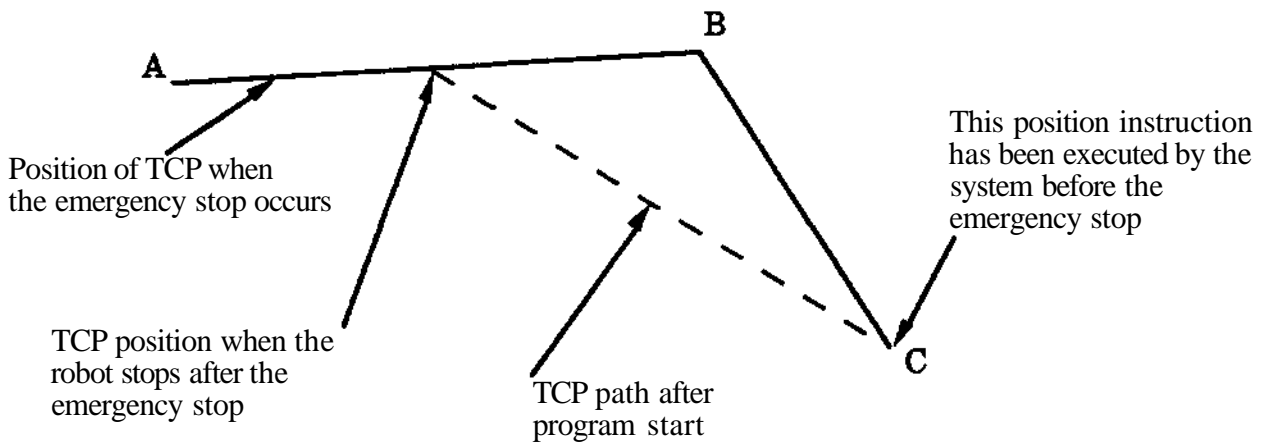
Backward movement
to B after start



3 Movement principles



Observe that if the velocity is high and/or the programmed points are close, the instruction execution can be several steps ahead of the physical position of the TCP. This can result in an unexpected path at Program Start after emergency stop or similar.



To avoid collision risk in these situations, the Move Restart function (see 7.10) should be used instead of Program Start.

Note: Emergency stop or similar connected as "Category 1 safety stop" (controlled stop), see Installation S3 chapter 3.5, does not cause this phenomenon.

3.6.2
Speed optimization

Speed optimization is advantageous in applications where problems are caused by speed reductions. Using speed optimization, the movement characteristics will be similar to that of ABB robots of M87 model or earlier. This means, for instance, that there will be no speed reductions due to overlapping corner zones and that the cycle time will be shorter than for path optimization. Speed optimization is only described in 3.6.2. In the rest of this manual the default value, path optimization is presupposed.

The zones used at corners have different meaning and values, depending on which optimization principle is used. Path optimization uses a corner zone, showing how far ahead of a point the control system will begin to generate a parabolic path. Speed optimization uses a zero zone, showing the servo system how close to the point the axes should be before driving towards the next point. Both types of zones are installed with MANUAL/PARAM/CHANGE/AUTO/ZONE. If path optimization is chosen, corner zones will automatically be installed and if speed optimizing is chosen, zero zones will be installed.

3.6.2.1 Guide text

The below tables show the guide texts used for zero and corner zones respectively.

Instruction texts:

Speed optimization	Path optimization
POS FINE	POS FINE
POS FINE L	POS C1
POS FINE XL	POSC2
POS	POS PATH

Zone installation texts:

Speed optimization	Path optimization
FINE	ZONE
SMALL	FINE
LARGE	CORNER 1
XLARGE	CORNER 2

3.6.2.2 Programming

When speed optimization has been chosen, three different types of fine points and coarse points can be chosen.

- Fine point with SMALL, LARGE or EXLARGE zero zone. The zero zone is defined with the installation parameter PARAM/CHANGE/AUTO/ZONE.
- Coarse point without defined zero zone. You can however define a zero zone also for coarse points. The coarse point will then work as a fine point.

The accuracy of a programmed point is always the same, independent of whether it is a coarse point or a fine point, and independent of the size of the zero zone. The difference is that with coarse points the next instruction will start when the robot begins to decelerate towards the point. If the next instruction is a WAFT, the robot will continue to the point as if it was a fine point. For fine points the next instruction will begin when the robot is within the zero zone for the point.

If you want to use speed optimization for programs programmed for path optimization, it might be necessary to adjust some points in the program. Also extra points and wait instructions could be used to obtain the requested path.



Use of coarse points without defined zero zone can cause the same problem as emergency stop in connection with path optimization. See 3&1.5 for more information.

3.7 Circles

Circles are programmed by putting in a circle point between two ordinary positions. The circle point is used only to define a position on the circle, i.e. speed and orientation are not influenced by the circle point.

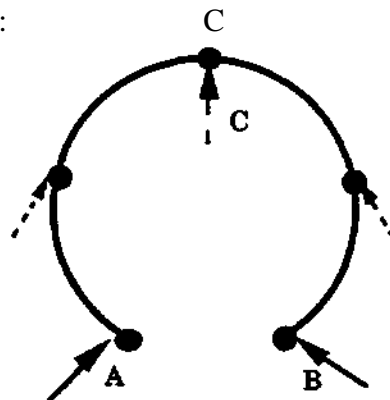
3.7.1 Orientation interpolation

There are three different ways to interpolate the wrist orientation in a circular movement:

Interpolation with CIRCLE = 0

The wrist orientation will be interpolated as during a linear movement. All reorientation calculations are handled as orientations relative to the base coordinate system. This means, that the wrist orientation will change from start pos A to end pos B, without respect to the TCP's circular movement. As in linear interpolation, the wrist reorientation will always be done in the shortest possible way, i.e. reorientation during path will always be less than 180 degrees.

Example:



Interpolation with CIRCLE = 1

The wrist orientation will be interpolated as orientations relative to the circle coordinate system (CCS). This means, that the wrist orientation will change from start pos A to end pos B, with respect to the TCP's circular movement.

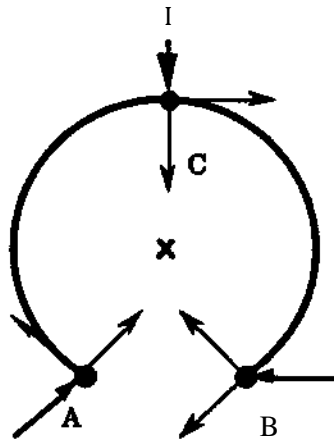
The CCS is a coordinate system which moves along the circle line during interpolation.

As in linear interpolation the wrist reorientation will always be done in the shortest possible way, i.e. reorientation during path will always be less than 180 degrees (in the CCS).

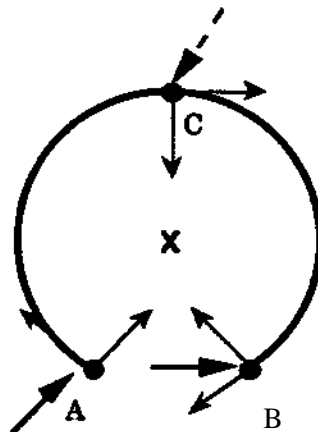
3 Movement principles

Observe: Only CIRCLE = 1 should be used when running circles in a rotational EXTFRAME.

Example with small reorientation relative to the CCS:



Example with big reorientation relative to the CCS:

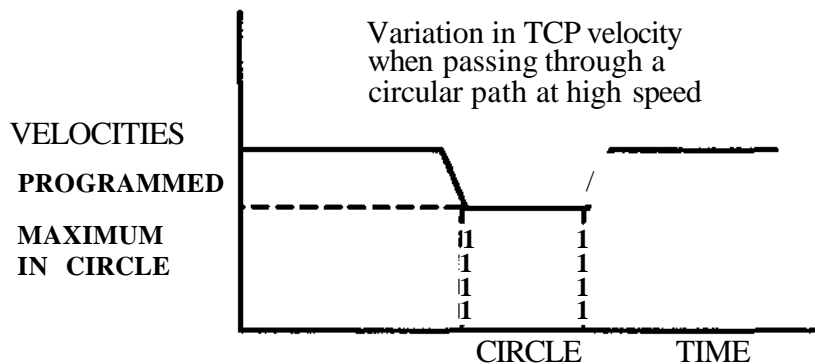


Interpolation with CIRCLE = 2

This mode is intended only for special needs and is normally not used. For small reorientations, it is identical to CIRCLE = 1. For big reorientations, the reorientation will normally be made in the opposite direction, i.e. the longer way.

3.7.2 Velocity in circle.

The maximum available velocity in a circle is a function of the circle diameter. It can also be limited by large wrist reorientation.



Maximum attainable velocity in a circular path (without large wrist reorientation)

circle radius	maximum speed limited by robot	maximum speed to achieve
2,5 mm	90 mm/s	10 mm/s
5mm	130 mm/s	20 mm/s
12 mm	220 mm/s	50 mm/s
25 mm	300 mm/s	100 mm/s
50 mm	450 mm/s	200 mm/s
100 mm	650 mm/s	400 mm/s
200 mm	900 mm/s	800 mm/s
400 mm	1200 mm/s	1200 mm/s

3.8

Soft servo

3.8.1

Introduction

The SOFT SERVO function only exists in the MH/GI/SW software.

The SOFT SERVO function enables the operator to:

- define 9 sets of "softness percentages"; each set containing a softness value for each axis
- manually activate any of the 9 sets, or normal axis control
- program an instruction in the robot program for activation of any of these sets, or normal control
- check the number of the active "softness percentage" set.

When SOFT SERVO for any axis (1-7) is active, a force applied on this axis will cause a positional deviation from the programmed position. The deviation is directly proportional to the force (except for frictional effects which are significant for small forces).

3.8.2

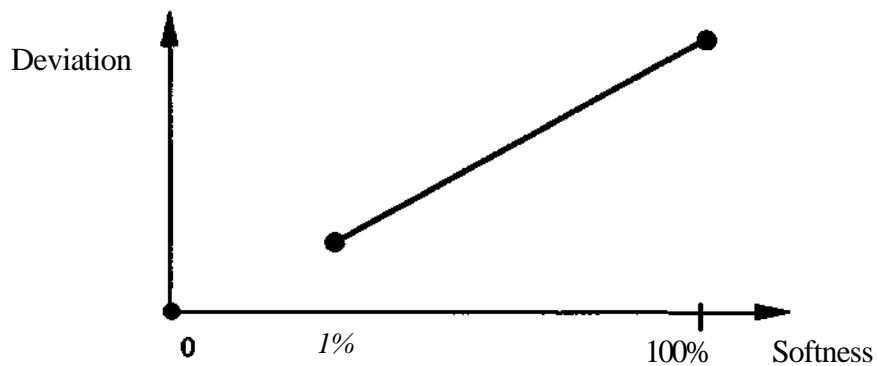
Definition of softness

When the "softness" for an axis has the value 0%, the axis is controlled normally and no deviation is allowed. If a force is large enough to cause deviation from the programmed position, an error message is displayed and program execution is stopped.

When the axis is "soft" with percentage value between 1-100, an outer force will cause axis deviation.

Static loads are automatically compensated, thus the robot will not move at activation or deactivation of SOFT SERVO, if the load remains the same.

For a specific face, the deviation will increase for increased softness, see fig below



3.8.3

Use of SOFT SERVO

SOFT SERVO is a software alternative to mechanical compliance in grippers in applications where imperfections in processed objects can be assumed. Assembly, grinding and polishing are such applications.

SOFT SERVO can also be used to limit the forces caused by the robot on the processed objects, e.g. assembly of fragile parts such as glass.

"Trial and error" methods in assembly applications can be performed with SOFT SERVO together with the function "JUMP IF NOT INPOS". This function controls the program depending on whether the robot has reached close enough to the programmed position or not.

Different characteristics of SOFT SERVO compliance can be used in the same robot program.

Soft servo is defined for each axis individually. The purpose is often to obtain "softness" in a specific rectangular direction and therefore the softness should be defined on those axes that mainly are working in that direction. Because of the robots non-rectangular structure some softness is also obtained in other directions.

Often, an application for soft servo requires a robot TCP movement in one direction while a force is applied at the TCP in another direction. A robot configuration should be chosen so that the main movement is carried out by one or two axes and these should be given a rather "stiff" softness value (for example 10%), while the force is taken up by one axis (with movement in the direction of the force) and with a "soft" value of soft servo (e.g. 70% to 90 %). In other words, softness in the axes contributing the most to the TCP movement should be avoided.

The brakes are activated 30 seconds after the robot has stopped during program execution. Activated brakes disable softness. If a long period with force on an axis without movement is required, activate softness immediately after movement has ceased. In this case, SOFT SERVO will be active for 9 hours until brakes are activated.

Some notes regarding the SOFT SERVO function:

- * if SOFT SERVO is active for any axis from 1 to 7 the dynamic performance of all axes is decreased
- * the static load compensation is the same as was valid in the position, when SOFT SERVO was activated. When the "soft" axes are moved from this position, they will deviate from the nominal path. This will result in axis movement when the function is deactivated as the nominal part in resumed.
- * Motion speed affects the path strongly when SOFT SERVO is active. Using delays (for example "WATT 0.5 S") helps keeping the path close to nominal.



When using the Soft Servo function, speed and path are affected!

3.9

Program displacement

Two different functions are available for program displacement:

- Parallel displacement by means of a reference point.
- Program displacement by change of reference frame, i.e. displacement of the world coordinate system.

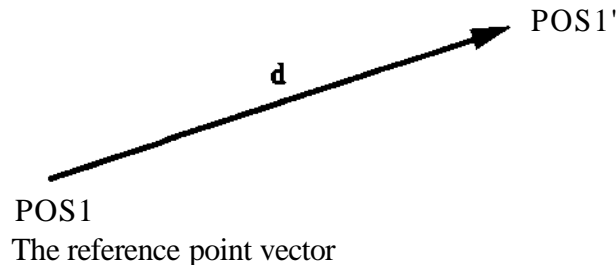
Both functions mean that all positions in a program are displaced in relation to their original locations. Both functions should be activated at a FINE position.

3.9.1

Parallel displacement with a reference point (REFP)

Parallel displacement is used for temporary displacement which will change for every new operation cycle of the robot. A typical application is when the exact location of an object is being searched for with a sensor, and the following sequence of positions should all be displaced equally.

Only one reference point can be defined at a time and it is always defined and activated in the same instruction, POS KEFPOINT. When programming the instruction POS REFPOINT the position POS1 (see the figure below) is stored in the instruction. When executing the instruction the robot is in position POS1' after a search operation. The REF POINT vector (d) defines the displacement of the Object Coordinate System (figure in section 3.2.1) relative to the World Coordinate System. If no FRAME is active the World Coordinate System will be the same as the Base Coordinate System.



3 Movement principles

Example

Subprogram 5 includes a movement pattern which must be capable of parallel displacement. For practical reasons, the movement pattern is placed in a separate subprogram 100.

Program 5:

200 POSV = 50% FINE

The position for the reference point is defined here. This position would typically (but not necessarily) be a search position.

210 CALL PROG 100

Execution of program 100 then continues from this position.

Program 100:

10 POSV REFPOINT ON

The reference point is activated in the position of the robot when instruction 200 in program 5 was executed

20 POS V = 100 %

The movement pattern (including other instructions) is performed parallel but displaced

150 POS V = 100 % FINE

Movement to the final point in the pattern

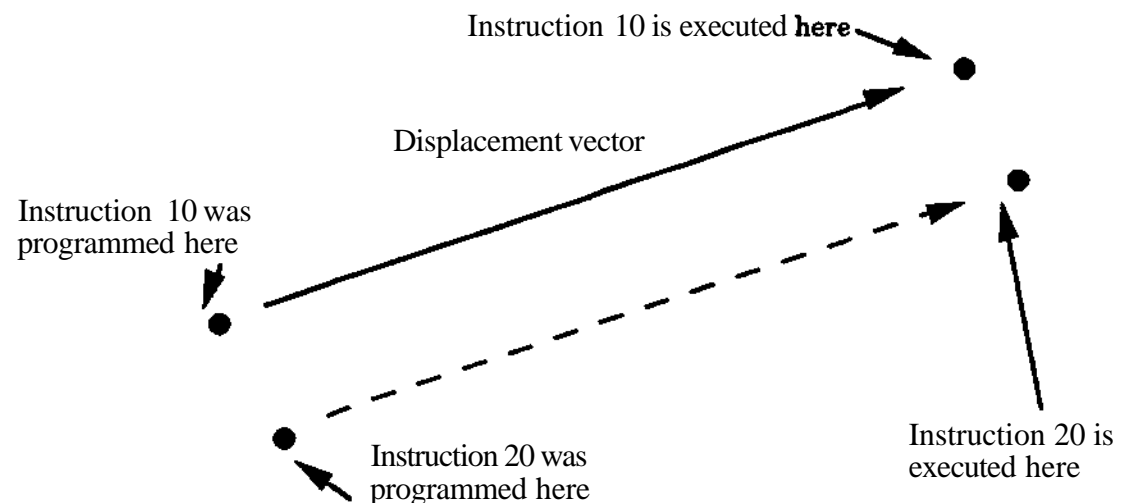
160 POS V = 0 % REFPOINT OFF

The reference point is deactivated.

170 RETURN

Program execution returns to subprogram 5.

When the instruction REFPOINT is executed a displacement vector is calculated which is equal to the difference in position between the point where REFPOINT is executed and the point where it was originally programmed. All points after REFPOINT will be displaced in accordance with this displacement vector. See the figure below.



Note!

The instructions 10 and 160 are not positioning instructions. Their only purpose is to activate and deactivate parallel displacement of the program section between them.

3.9.2

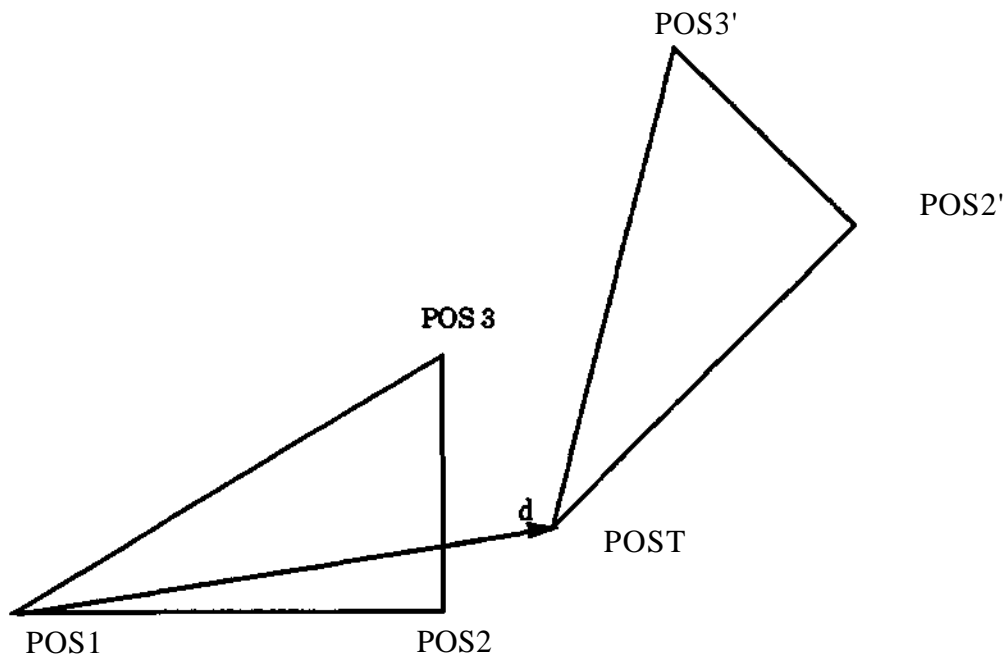
Program displacement with reference frame (FRAME)

With a FRAME instruction it is possible to displace and reorient all positions in a program with the same amount. FRAME is used if a more permanent change has occurred in the station, (e.g. the robot has been displaced or a fixture has been moved). The FRAME function can, however, also be used in combination with search functions to adapt to temporary displacements of the object, like REF.POINT. The FRAME function is more powerful than the REF.POINT, as it also allows for rotations, but it is more complicated to define.

All stored positions in position instructions are expressed relative to the World Coordinate System WCS (let us suppose that REF.POINT is zero) (see the figure in section 3.2.1). If there is no FRAME active the WCS will coincide with the Base Coordinate System BCS. A FRAME instruction however will displace the WCS relative to the BCS with an x, y, z value and a quaternion value for reorientation (both symbolised by vector FRAME) i.e. the WCS will be displaced **and** rotated relative to the BCS. Rotation may be performed around any axis.

Up to five FRAME-vectors may be stored in frame registers in memory, i.e. FRAME 1 - FRAME 5. A FRAME-vector can be defined by dislocation of three positions (see figure below). This can be done either manually by moving the robot with the joystick or automatically during program execution. A FRAME vector is stored in the frame register until it is overwritten by a new definition of the frame with the same number.

If possible, the orientation of the tool should be kept unchanged, to get the highest accuracy.



DEFINING A FRAME

3 Movement principles

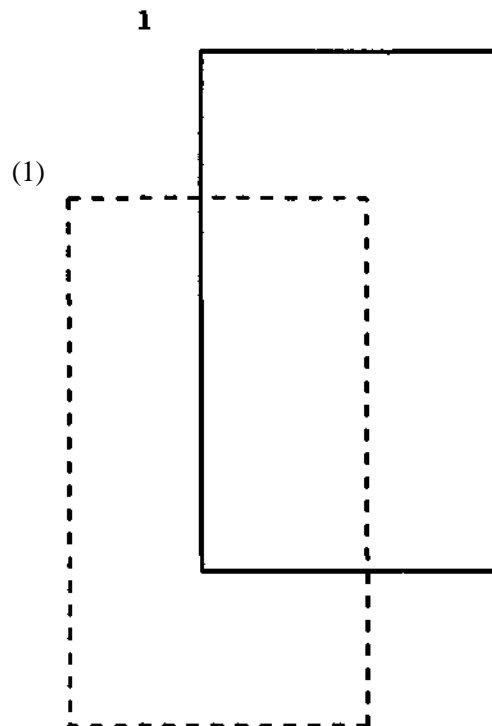
When defining a frame in automatic mode, the dislocations are measured with POS FRAME POS instructions. Suppose that the robot is in position POS1 when programming the POS FRAME POS 1 instruction (see figure above). The coordinates for this position will be stored in the instruction. When executing the instruction the robot will be in position POS1', perhaps after a search operation, and the dislocation vector d will be calculated. The same operations are performed for POS2 and POS3. After this the execution of a FRAME 1 DEFINE instruction will calculate the FRAME vector from the three dislocation vectors and store the result in FRAME 1 (the same will go for FRAME 2-5). Please observe that no real dislocation of the following stored positions is performed yet. This will not happen until FRAME 1 is activated with a FRAME 1 instruction.

A new FRAME may be activated while a former one is still active. This will give an accumulated displacement. This should however be avoided since the accuracy will be suffering. Thus an active FRAME should be deactivated a FRAME 0 instruction before a new FRAME is activated. Also observe that through both FRAME instructions and POS REFPOINT instructions can be active at the same time.

Three positions are included in the definition of a program displacement. In order to succeed with the definition, the three positions must be located as below:

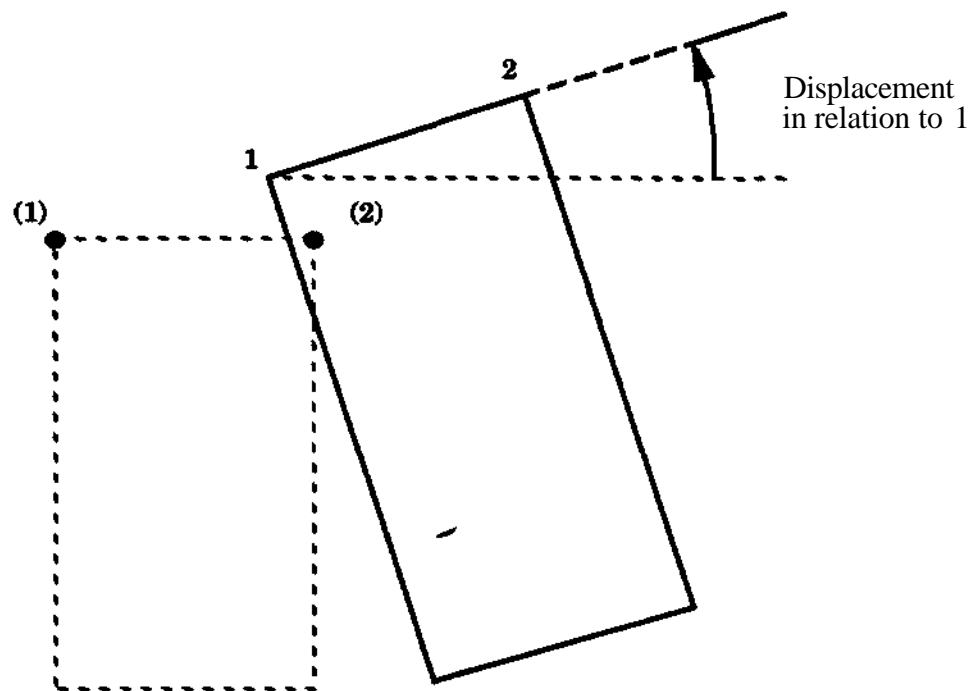
- In a triangle.
> 100 mm from each other.
- Position 1 must be defined with the high accuracy.
- Position 2 defines the direction of the line between POS1 and POS2 and can be positioned anywhere on the line.
- Position 3 defines the orientation of the plane through POS1, POS2 and POS3 and can be positioned anywhere in the plane. See figures below.

Position 1 defines the new location for position (1):

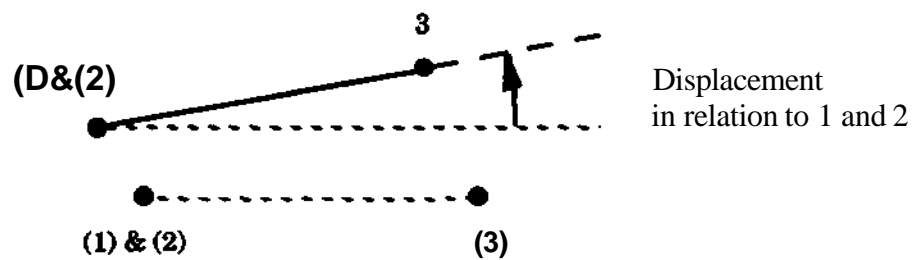


3 Movement principles

Position 2 defines the new direction, for the line (1) - (2), as 1 - 2:

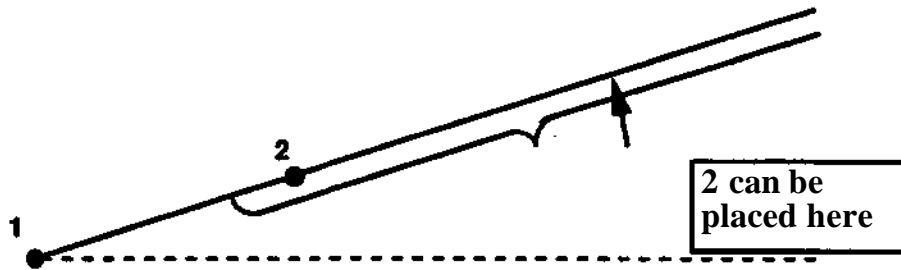


Position 3 defines the new position of the plane (1) - (2) - (3), as 1 - 2 - 3:

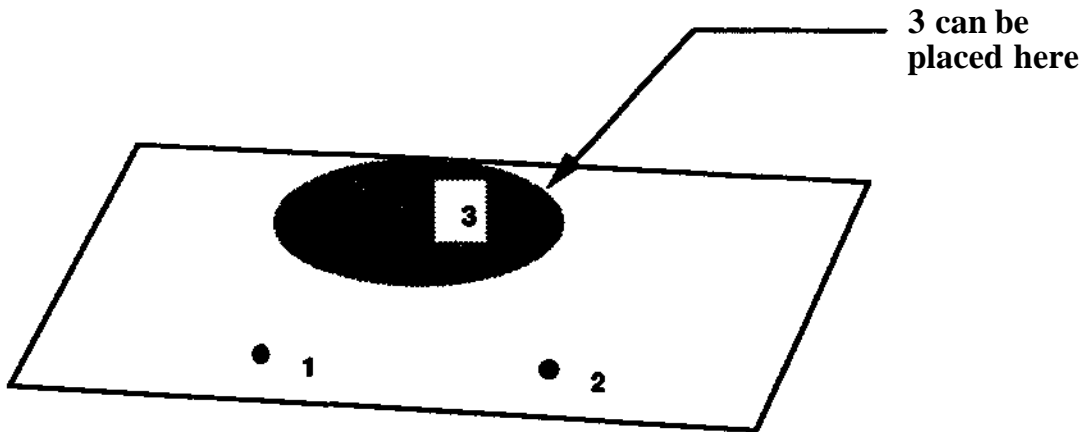


3 Movement principles

As the direction only is important, position 2 can be placed as in the figure below:



As the position of the plane only is important, position 3 can be placed according to the figure below:



3.10 Singular points

Singular points are points in the robot working range where the axis of rotation of two robot axes coincide. Sometimes, when driving the robot in rectangular coordinates, this will cause very big robot axes movements or movement stop (with error message). To overcome this problem when jogging with the joystick, the robot coordinate system should be chosen when the point is to be passed.

In the case of program execution, singular point problems can be handled by the following instructions:

MODRECT COORD enter the modified rectangular coordinate system before the positioning instruction at which the problem occurred. This means that linearity and speed are preserved but the wrist orientation may change.

RECT COORD return to the rectangular coordinate system after the problem area.

Of course **MODRECT COORD** can be used throughout the program if the orientation is of minor importance.

Another type of singularity occurs when the wrist rotates around the z-axis of the Base Coordinate System. To meet this, the following limitations are introduced:

- The wrist center can not move through a prohibited zone with the radius 2 cm in rectangular coordinates. A pass-through can be performed in robot coordinates.
- When running in **MODRECT** coordinates, the TCP as well must not be located inside a corresponding prohibited zone around the z-axis. The radius of the zone in question varies and can as a maximum reach a size equal to the TCP vector, depending on the TCP orientation at the pass-through.
- It is not possible to program a point with the wrist center inside the prohibited zone.

Section	Page
4.1 Start	4:3
4.1.1 Power on, power off, synchronization	
4.1.2 Program Execution	
4.1.3 Stop	
4.2 Restart after power failure	4:5
4.2.1 General	
4.2.2 Automatic restart	
4.2.3 Manual restart	
4.2.4 Restart from PLC	
4.3 Language selection	4:8
4.4 Software configuration	4:8
4.5 Program information	4:9
4.6 Error buffer	4:10
4.7 Disk handling	4:11
4.7.1 Initialize disk	
4.8 Computer link	4:13
4.8.1 General	
4.8.2 Commands to robot	
4.8.3 Commands from robot	
4.8.4 Spontaneous status messages	
4.8.5 Request for superior control	
4.8.6 Operation status	
4.9 Register	4:15
4.9.1 Register	
4.9.2 Position registers, location	
4.10 Inputs and outputs	4:17
4.11 Ports	4:17
4.11.1 Digital ports	
4.11.2 Analog ports	
4.12 SYSTEM-I/O	4:22
4.12.1 Standard SYSTEM-I/O, inputs	
4.12.2 Standard SYSTEM-I/O, outputs	
4.12.3 Additional SYSTEM-I/O at I/O MAP	
4.12.3.1 Summary of HOLD, HOLD RESET and AUTO INPUT	
4.12.3.2 Inputs	
4.12.3.3 Outputs	
4.13 Remote control panel signals, panel- I/O	4:28

4.1 Start

4.1.1 Power on, power off, synchronization

Power on



Before the robot is started, ensure that the working range of the robot is unobstructed and that no person is in any dangerous zone.

The robot must be 'left alone' during the initiation.

The initiation is ended by displaying the text shown below at the same time as all the panel and programming unit lights are lit for 3 seconds.

"ABB ROBOT SYSTEM AT YOUR SERVICE"
SOFTWARE FOR IRB XXXX

After the initiation followed by MOTOR ON command a safety test is performed. This will cause the robot movement, initiated by a program start or by the joystick, to be delayed about 10 sec.

Power off

First, stop any program execution. Then bring the robot to the MOTOR OFF mode. Turn off the mains switch.



When the robot is switched off, all outputs are reset. This may affect the peripheral equipment and grippers which in turn might cause damage.

The memory utilizes battery backup, and is not affected by mains switched off.

Synchronization

If optional external axes are not equipped with absolute measurement system, the MOTOR ON lamp will start flashing when the robot is taken to the MOTOR ON mode for the first time after initiating. This is to indicate that the robot has not been synchronized.

Press the programming unit SYNC button to synchronize the external axes. The MOTOR ON lamp will be on steadily when the synchronization is completed.



When SYNC is selected, the external axes immediately start moving. Make sure nobody is inside any dangerous zone.

4.1.2 Program Execution

Programs can be executed only in the MOTOR ON mode.

Position instructions can be executed

- Instruction by instruction forward
- Continuously forward
- Instruction by instruction backwards

Other instructions can only be executed forwards, instruction by instruction or continuously. When executing backwards, other instructions have to be skipped, using the SKIP BW button.

When a programmed movement is executed, the robot always moves towards the programmed position irrespective of where the movement began.

When the operating mode selector is in any of the positions MANUAL REDUCED SPEED or MANUAL FULL SPEED, built-in safety functions ensure that:

- The robot can not be controlled via the remote control or computer link.
- The enabling device must be operated to switch to the MOTOR ON mode.
- The button starting the execution must be kept depressed (HOLD TO RUN)

When the operating mode selector is in the MANUAL REDUCED SPEED position, the following also applies:

- The robot programs are executed at max. 250 mm/s.
- The HOLD TO RUN-function can be deactivated by a system parameter

4.1.3 Stop

Program execution may be stopped in several ways:

- manually, by releasing HOLD TO RUN
- manually, using the programming unit STOP button.
- manually, using the remote control STOP button.
- automatically, using the program stop digital input (system I/O)
- automatically, using programmed stop in the robot program
- automatically, due to a system fault

Note!

Program execution stop performed in one of the above mentioned ways results in a shortest possible smooth stop on the path. Release of the enabling device in MANUAL mode or use of any switch in the MOTOR ON chains will activate the brakes immediately causing the robot to stop without path control.

Note!

Stopping program execution in the MANUAL REDUCED SPEED and MANUAL FULL SPEED modes should be done by pressing the PROG STOP button on the programming unit before releasing the safety pad. Otherwise the braking is done more harshly since hardware is performing the braking (safely demand).

NandR

On program stop, an R (ready) is shown at the right hand side of the second line of the display if the current instruction is completed. An in completed instruction is shown by the letter N (not ready). (An instruction can be executed twice by entering the Editing menu between the two executions).

4.2 Restart after power failure

4.2.1 General

By using the automatic restart function program execution can be resumed at the instruction interrupted by a power failure. Automatic restart can be performed at:

- the programming unit
- a remote control panel
- a superior computer

A robot system provided with absolute measurement need not be synchronized before restart, as the measurement system automatically registers the robot position **anywhere inside the working area**. Calibration of the measurement system is performed during installation. See the Installation S3.

4.2.2 Automatic restart

When automatic restart is possible, the following apply:

- The MOTOR ON lamp is flashing
- The text RESTART POSSIBLE is shown on the display of the programming unit, together with soft key texts CH OUT (check outputs) and NO RSTRT (no automatic restart).

One of the following four alternatives is selected:

1. MOTOR ON command is issued
 2. Press CH OUT on the programming unit
 3. Press NO RSTRT on the programming unit.
 4. Restart from PLC i.e. by remote panel, see chapter 4.2.4.
1. This commences a procedure for a simple automatic restart without output checks:
 - 1A MOTOR ON command
 - 1B RESTART command
 - 1C START PROGRAM or MOVE RESTART or AW RESTART command. The robot will then continue the program where the interruption occurred.
 2. This allows checking output values before the automatic start is activated:
 - 2A press CH OUT (check outputs) on the prog. unit.
 - 2B enter the logical number of the output to be checked.
 - 2C CHANGE the output value, or check the next output
 - 2D when outputs are checked press BREAK and the MOTOR ON button.
 - 2E Press RESTART on the programming unit. The start can be aborted by pressing NO RSTRT (no restart) on the prog unit.
 - 2F Press the START PROGRAM button or MOVE RESTART or AW RESTART button on the programming unit.
The robot will then continue the program where the interruption occurred.

3. This allows a manual restart as described in the next section.

On pressing RESTART the first line of the Automatic Menu is displayed on the programming unit. The robot prepares automatic restart of the program. This means that the following functions will be set as before the power failure:

- Digital outputs. (As an alternative, outputs can be set to a status as explained above)
- All registers.

- TCP and FRAME.
- Reference point.
- The calling sequence between the different subprograms.
- The speed correction.

Note!

Analog outputs are cleared.

4.2.3 Manual restart

Manual restart is performed when:

- The system parameter for automatic restart has been deactivated.
- The operator selected NO RSTRT, i.e. manual restart on the programming unit.
- The robot is too far from the programmed path (error message displayed).
- Any of the buttons EDIT or MAN was pressed after program execution or the robot was moved manually by means of the joystick before the power failure occurred.
- The robot, or an external axis, is outside the working area (error message displayed).
- An instruction, which cannot be restarted was performed at the power failure (error message displayed).
- The internal close-down of program execution failed (error message displayed).

Manual restart means that:

- The first instruction in program 0 is presented.
- All outputs are reset.
- TCP 0 and FRAME 0 are activated.
- REFPOINT is deactivated.
- The absolute speed is set to 1000 mm/s and the speed correction to 100%.

Manual restart is performed as follows:

1. Go to MOTOR ON status.
2. Restart the program running from a suitable instruction in the main program.
All calling sequences has disappeared, caused by the power failure. Run the robot manually to a suitable position.
3. Check the status for digital outputs, and values in registers, and change if required.

When everything is ready, start program running.

Program start (PROG ST, INS ST, EXEC BW, AW REST) means that the robot will move at high speed.



Make sure nobody remains inside the robot working range and that all safety equipment is functioning correctly.

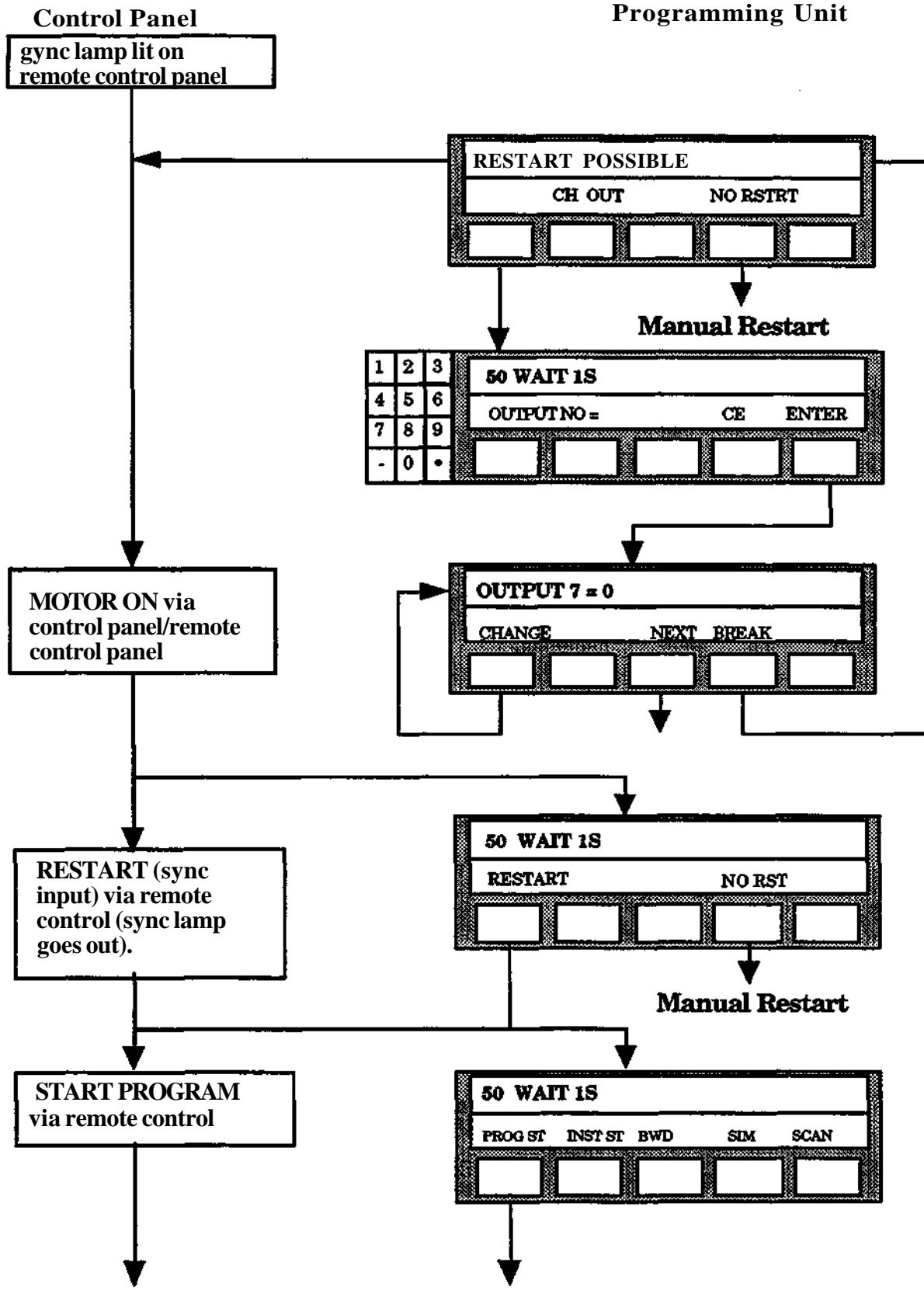
Before program execution is started make sure the robot is in a suitable position.

4.2.4**Restart from PLC**

After a power failure the logic for a pic can be like the following:

- Read output SYNC.
- When SYNC is high (after lamptest which is approx 3 seconds) then set input MOTOR ON with a pulse.
- Read the output MOTOR ON.
- When MOTOR ON is high, set input SYNC with a pulse.
- When output SYNC is low, start program execution by either PROG START or MOVE RESTART.

See chapter 4.12 and 4.13 for definition of the signals. (If MOVE RESTART is desired, I/O-map has to be used.)



4.3

Language selection

The dialog with the robot system can be performed in ten different languages: English, German, French, Dutch, Spanish, Italian, Portuguese, Finish, Japanese (katakana) or Swedish.

The language used can be changed, at any time during the course of the dialog. Not only the menus, guide questions and messages presented on the display are changed, but all of the robot programs are presented in the new language.

Language is changed according to the following:

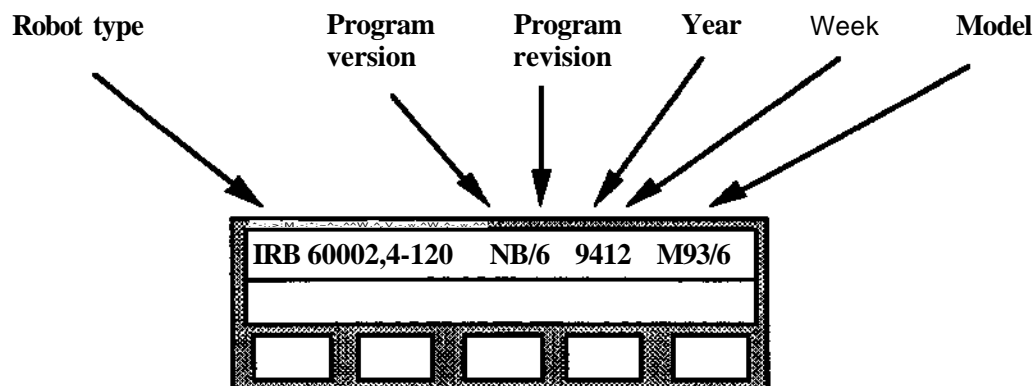
1. Put the robot system in the MOTOR OFF mode.
2. Press the MANUAL button.
3. Press the SCAN button until a menu containing an asterisk (*) is shown.
4. Press the button directly under the asterisk.
5. Select the language from the menu.

The asterisk is there to simplify finding the language function, no matter if the language presently on the display is not understood by the operator.

4.4

Software configuration

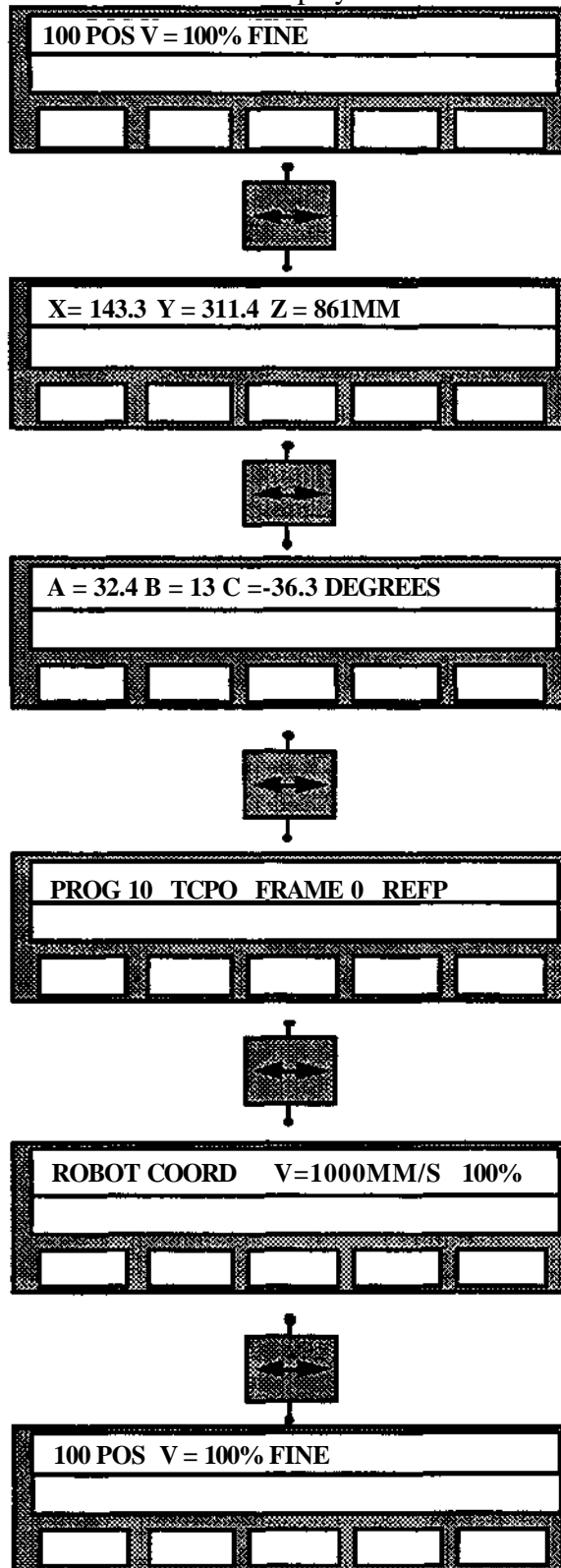
The code for the software supplied in the controller can be read on the programming unit display or on the Monitor. When the programming unit display shows ERRORS, these should be scanned through by pressing the SHIFT button until finally details of the software configuration are shown.



4.5

Program information

Program information is available by using the SHIFT button. The following example shows what information is displayed:



Euler angles for the
programmed position.

4.6 Error buffer

The system contains an internal error buffer, which can store 9 error messages. In the buffer the system stores:

- All kinds of error messages (both system errors and operator errors), occurred at the latest error occasion.
- Earlier messages about system errors.

The system can, on request, present the contents of the error buffer via:

- The programming unit. (Just error messages from the latest error occasion.)
- A printer, if the optional function Program printout is provided.
- A monitor, if this optional function is provided.

Display on the programming unit of error messages.

The programming unit displays one error message at a time, according to the following:

- When the system stops running due to an error, the operator will see the first error message. An arrow on the display indicates if there are any consequent errors. In this case the operator can display these messages also, one by one.
- During manual operation the operator can, on request, display stored messages from the latest error occasion, one by one. The error messages are chronologically displayed.
- For most error messages, it is possible to get text in plain language by pressing "•" on the programming unit.

Display on the monitor of error messages.

The monitor displays all the messages within the error buffer, according to the following:

- When the system stops running due to an error, the operator will see all messages in plain language, occurred during that error occasion. The error messages are chronologically displayed.
- During manual operation the operator can, on request, display all stored messages. The error messages are chronologically displayed. The oldest message on top of the list.

Printout of error messages.

- During manual operation the operator can, on request, display all stored messages. The error messages are chronologically displayed, the oldest one on top of the list.

Erase of the contents in the error buffer and display of text in plain language.

- During manual operation the operator can, on request, erase all messages stored in the error buffer.
- On request, the operator can get text in plain language for most error messages.
- On request, the operator can load the texts in plain language from disk (See Installation S3).

4.7

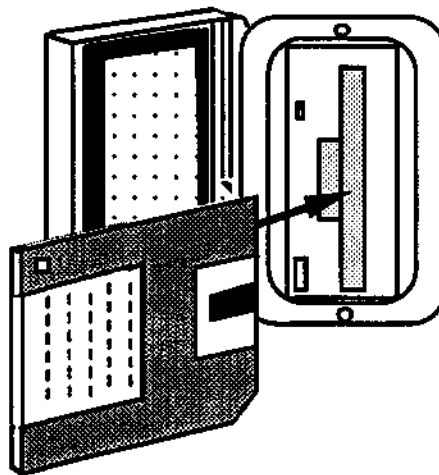
Disk handling

The disk drive is used for storage and loading of robot programs, system parameters and program parameters. 3.5"-2DD (two side, double density) 1Mbytes disk are used.

A block corresponds to the whole contents of the user memory, including main program, all subprograms and program data.

Up to 7 program blocks with different numbers can be stored on each disk, in addition to system parameters, and program parameters (See Installation S3 for system parameter information).

Program blocks can have numbers from 0 to 9999. Program blocks stored previously are written over if a new block with the same number is entered for storage.



4.7.1

Initialize disk

A new disk must be initialized before being used. The contents of a disk used previously will be erased by initializing the disk again.

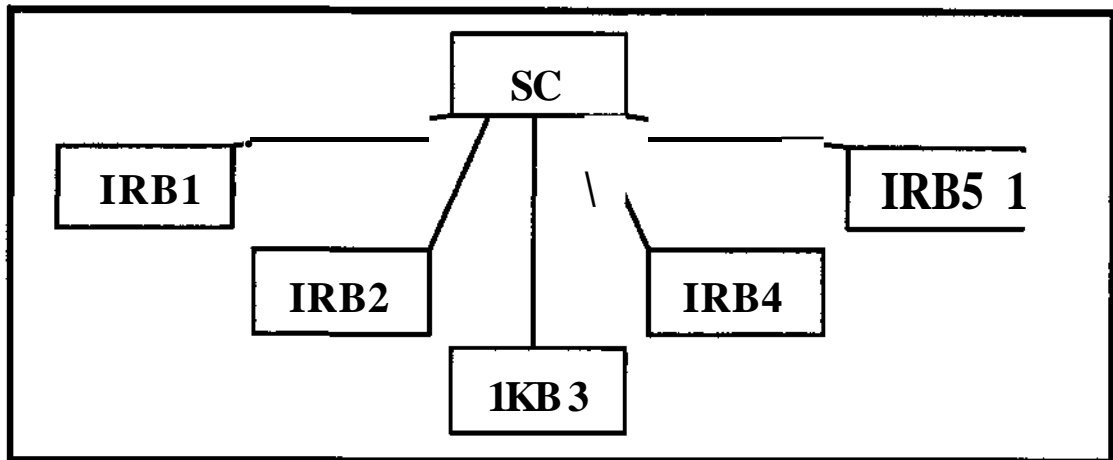
A disk can only be initialized when the robot is in MOTOR OFF mode.

The procedure is described in section 9.2.3.

4.8 Computer link

4.8.1 General

Connections to the Superior Computer are point-to-point as shown below.



4.8.2 Commands to robot

A large number of commands can be given to a robot from the superior computer (SC). Each command must be acknowledged by the robot. The different types of commands are described below:

- Loading of program blocks or individual programs from the SC.
- Start or stop of program execution.

Reading and writing of

- robot data
- process data

Reading of the current status of the robot

- Position of active TCP, wrist and external axes
- Change of robot mode (MOTOR OFF, MOTOR ON etc)
- Control of IRB from S.C. by means of X, Y, Z and wrist position and orientation coordinates.

Note: While the robot program is executing logical (i.e. non-position) instructions, commands from SC can not be received. Therefore if a long sequence (approximately 15) of logical instructions is executed, e.g. in a loop waiting for an input, occasional WAIT 0.1s instructions should be entered so as to allow receptions of SC commands.

4.8.3

Commands from robot

A few commands can be initiated manually from a robot to the SC via the programming unit:

- Storage or loading from the SC of program blocks or an individual program (for arc welding robots even weld data, see chapter 12).
- Storage or loading from the SC of system data.

4.8.4

Spontaneous status messages

When certain events occur in the robot, a message is automatically transmitted to the superior computer. The message contains information regarding the current robot status with respect to:

- Robot mode (MOTOR OFF, MOTOR ON, PROGRAM EXECUTING or EMERGENCY STOP).
- The operating mode selector in position AUTO or MANUAL REDUCED SPEED/MANUAL FULL SPEED.
- Interrupt via direct acting inputs (permitted or not permitted).
- Active program and instruction number
- Position of active TCP, wrist and any external axes.

A spontaneous status message is transmitted each time:

- The instruction SUCTRL is executed. See section 4.8.5
- Emergency stop is tripped.
- Search stop is given.
- Program execution is stopped manually.
- The operating mode selector is switched between MANUAL REDUCED SPEED and AUTO (not between MANUAL REDUCED SPEED and MANUAL FULL SPEED).
- Faults which activate error codes occur.
- The robot is powered on.

4.8.5

Request for superior control

The robot will send a request for superior control to the SC by executing the instruction SUCTRL. The instruction can have a combination of functions for program stop and register information.

4.8.6

Operation status

A robot can be run in two different operation modes:

- Local operation mode, when the robot control system does not accept commands from an SC.
- Remote operation mode, when commands from an SC are processed by the robot control system.

The different operation modes can be selected manually using the Programming Unit.

After power on, the remote operation mode is obtained automatically if the robot is configured for a computer link.

The robot can be programmed in either operation mode. It is however necessary for the local operation mode to be selected when programs are transferred to and from disk and when function parameters are changed. The remote operation mode must be selected when the above operations are to be performed with the SC.

All functions are allowed in the AUTO mode, but the following are not allowed in the MANUAL REDUCED SPEED and MANUAL FULL SPEED modes when issued from a S.C.

- | | |
|--------------------------------|--------------------------------|
| • Write to program/block | • Write to frame register |
| • Start of robot program | • Order MOTOR ON and SYNC |
| • Write to sensor register | • Write arc weld data |
| • Write parameters | |
| • Write to TCP register * | • Erase program |
| • Write to location register * | • Load program/block from disk |
| • Write to register * | • Write positioning data |
| • Write to digital output * | • Write to analog outputs |

*^E Allowed from control program M93/2.

4.9 Registers

4.9.1 Register

Control of program or peripheral equipment is performed on the basis of a value in a numerical register.

The system contains 120 numerical registers, number 0 -119, where one integer value at a time can be stored. When a new value is stored, the previous value is written over. Permitted values range from - 32 768 to +32 767.

During program execution, a numerical value can:

- Be stored directly.
- Be stored indirect from a digital or an analog input port.
- Be sent to the peripheral equipment via a digital or on analogue output port.
- Compared with another value in a jump instruction.
- Be added, subtracted, multiplied or divided with another register value.

- Be a:
 - Program number at an indirect call of a program.
 - Position register number when running to a stored position.
 - Block number at indirect read of a block from diskette.
- Define a value for a robot position
- Specify which module in a pattern program is to be called.
- Define displacement or rotation of a location, or a position, defined by one of the instructions POS LOC, POS POS or RELTOOL (not on ARCW software).
- Define the current position in a pallet (not on ARCW software).

It is possible to manually check and change the value in one register at a time.

The register instruction is used for:

- Storing a value into a numerical register directly or via other registers.
- Fetching a value from a port.
- Transferring a value from a register to a port.
- Arithmetic handling of the register value.
- Entry of location coordinate values.

The manual function is used in checking and editing the program.

The robot reacts immediately as soon as a manual or a programmed command, as described above, is concluded.

4.9.2

Position register, location

Means:

Storage and reuse of a position in a position register during program execution as an argument in positioning instructions.

Facts:

When using a position register there is a difference between **location** and **position**.

- A robot **location** is only three coordinates in the memory, i.e. the TCP location. Positions of external axes are also saved.
- A robot **position** is, besides of the TCP coordinates, also the tool orientation and the positions of external axes (provided that any one of the position registers 0 - 95 is used).

(Arc welding robots only)

By activating/deactivating a parameter (REF POINT MODE) it is possible to compensate/not compensate for the influence by an active reference point.

Used:

- When the same position is to be used for the tool several times during a work cycle.
- When a programmed position is to be displaced step by step each time the tool moves to the position.

Executed:

By the STO POS-instruction, **the TCP location, the tool orientation and the positions for the external axes** are stored in a position register.

By the POS POS-instruction a position is programmed in such a way that the robot, during program execution, will move to a stored **position with the tool orientation stored**.

The **position** can be provided with an offset if required.

4 Operation

The offset can be given in three different ways:

1. Entered from the programming unit as x, y and z values when programming the POS POS instruction.
2. Entered by moving the robot with the joystick the distance wanted, when programming the POS POS instruction.
3. The x, y and z values are read from three numerical registers when the instruction is executed. When the POS POS instruction is programmed the number of the referred numerical registers are given. In this way the offsets can be changed dynamically while the program is running.

By the POS LOC-instruction a position is programmed in such a way that the robot, during program running, will move to a stored location **with the current tool orientation**. The **location** can also be provided with an offset if required.

4.10

Inputs and outputs

Means:

The robot and peripheral equipment communicate via inputs and outputs of both digital and analog types. The analog version is described under "Ports".

Facts:

A digital input or output can have one of two different statuses, "1" (voltage) and "0" (no voltage). Via the outputs, the robot transmits information and commands to the peripheral equipment and the corresponding information from this is received at the inputs. The setting of outputs to 1 or 0 is managed by special instructions during program execution and such an instruction applies until a similar but contradictory instruction is encountered during the execution.

A manual function permits:

- checking of the status of individual digital inputs.
- checking of digital outputs individually. It is also possible to make, an immediate correction when the status of an output is found out to be faulty.

It is also possible to invert an output and to send a negative or positive pulse of 200 ms duration via an output (short change of the current status of the output).

4.11

Ports

A port is a group configuration of digital inputs/outputs or a single analog input/output. It is possible, in a user program, by using REGISTER instructions (section 5.8) to connect register 0-119 to ports. The connection is temporary and is only active in connection with execution of REGISTER instructions. The normal functions of the digital inputs/outputs in other program sections is not affected.

The function permits:

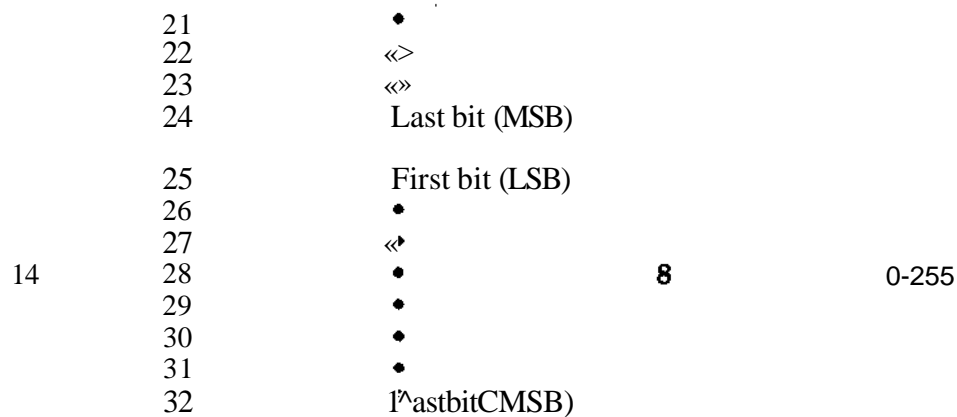
- The accessing of the status of a group of digital inputs for storage in a numerical register. The status of each of the inputs is given by a binary digit, "1" (closed input) or "0" (open input), and the binary value formed is stored in the register.
- Transmission of the binary value in a register to a group of digital outputs for setting these to one or zero. The value is coded in binary form, "V" (active output) or "0" (inactive output).

4.11.1 Digital ports

In this context a group is either 4 inputs/outputs (ports 1,2,11 or 12) or 8 inputs/outputs (port 3, 4, 13 or 14). There are two 16 bit ports, 70 and 80, which embrace both inputs and outputs. Two analog outputs, port 21 and 22, are used in arcwelding and glueing software, see chapter 12 and 13.

The group configuration of inputs/outputs for ports 1-4 and ports 11-14 are shown in the table below. (LSB means Least Significant Bit and MSB means Most Significant Bit".)

Port no.	Output no.	Logical output	No. of bits	Value in register
1	1	First bit (LSB)	4	0-15
	2	•		
	3	•		
	4	Last bit (MSB)		
2	13	First bit (LSB)	4	0-15
	14	•>		
	15	•>		
	16	Last bit (MSB)		
3	17	First bit (LSB)	8	0-255
	18	•		
	19	•		
	20	•		
	21	•		
	22	•		
	23	•		
	24	Last bit (MSB)		
4	25	First bit (LSB)	8	0-255
	26	•		
	27	•		
	28	•		
	29	•		
	30	•		
	31	•		
	32	Last bit (MSB)		
Port no.	Input no.	Logical input	No. of bits	Value in register
11	1	First bit (LSB)	4	0-15
	2	•		
	3	•		
	4	Last bit (MSB)		
12	13	First bit (LSB)	4	0-15
	14	•		
	15	•		
	16	Last bit (MSB)		
13	17	First bit (LSB)	8	0-255
	18	•		
	19	•		
	20	•		



The use of port 70 and port 80

The ports number 70 and 80 are used to transmit 16 bits binary values. The transmission is sequential and divided into four sections with four bits each time. The robot system and the peripheral equipment "shake hands" between each section.

Before the start of the transmission of the 16 bits, a control word of four bits is sent to the peripheral equipment. The value of this word, between 0 and 15, is determined by the register number. If a higher register number is used, the value will be Reg. no MOD 16, i.e. the remainder of an integer division by 16. E.g. Reg.no. 16= Control word 0, Reg.no 17= Control word 1, and so on.

In the user program it is only necessary to specify transferring or fetching of data to or from the different ports. The robot system then automatically manages all signal handling until the transmission is complete.

The table below shows the configuration of inputs/outputs for ports 70 and 80. The number of bits is 4 x 4, and the value in the register can be between -32.768 and +32.767.

Port no.	Logical in/output	Note	
Output no.			
70	1	First bit (LSB)	Outputs 1-4 used for control word and data
	2	•	
	3	•	
	4	Last bit (MSB)	
	5	•	Outputs 5-6 used for handshaking during data transmission
	6	•	
Input no.			
	5	•	Input 5 is used for handshaking during data transmission
Input no.			
80	1	First bit (LSB)	Inputs 1-4 used for data
	2	•	
	3	•	
	4	Last bit (MSB)	
	5	•	Input 5 is used for handshaking during data transmission

4 Operation

Output no.		
1	First bit (LSB)	Outputs 1-4 used for control word
2	•	
3	•	
4	Last bit (MSB)	
5	•	Outputs 5-6 are used for handshaking during
6	*	data transmission

There is a time supervision in the robot. If input 5 is not inverted within 400 ms after output 6 is inverted the program execution will be stopped.

When transmitting via Port 70 it is never checked that data bit 0-3 has been read input 5 inverted. Therefore a waiting time in the program will secure that data has been transmitted correctly before output 1-4 is changed.

Activity sequence at data transfer to port 70

Robot	External equipment
Set output 6	
Transmit control data to outputs 1-4	
Invert output 5	Wait 2 ms Read the control word Invert input 5
Transmit bits 12-15 to outputs 1-4	
Invert output 6	Wait 2 ms Read data Invert input 5
Transmit bits 8-11 to outputs 1-4	
Invert output 6	Wait 2 ms Read data Invert input 5
Transmit bits 4-7 to outputs 1-4	
Invert output 6	Wait 2 ms Read data Invert input 5
Transmit bits 0-3 to outputs 1-4	
Invert output 6	Wait 2 ms Read data (Invert input 5)

Activity sequence at data fetching from port 80

Reset output 6	
Transmit control data to outputs 1-4	
Invert output 5	Wait 2 ms
	Read control word
	Transmit data, bit 12-15 to inputs 1-4
	Invert input 5
Read data from input 1-4	
Invert output 6	Wait 2 ms
	Transmit data, bit 8-11 to inputs 1-4
	Invert input 5
Read data from inputs 1-4	
Invert output 6	Wait 2 ms
	Transmit data, bit 4-7 to inputs 1-4
	Invert input 5
Read data from inputs 1-4	
Invert output 6	Wait 2 ms
	Transmit data, bit 0-3 to inputs 1-4
	Invert input 5
Read data from inputs 1-4	
Invert output 6	
Reset outputs 1-4	

4.11.2**Analog ports**

For analog input/output signals there are four ports , numbered according to the table below. The same REGISTER instructions as for digital ports apply.

Port no.	Output no.	Limit values	Corresponding Register values
21	1	0± 10 V	0± 1024
22	2	0± 10 V	0± 1024
23	3	0± 10 V	0± 1024
24	4	0± 20 mA	0± 1024

Port no.	Input no.	Limit values	
31	1	0± 10 V	0± 1024
32	2	0± 10 V	0± 1024
33	3	0± 10 V	0± 1024
34	4	0± 10 V	0± 1024

4.12 SYSTEM-I/O

System-I/O is a group of functions which can be selected in two ways, either as a standard set according to the list below, or as a flexible set decided by the user (see Installation S3). The selection is done by defining the I/O USE - SYS parameter in PREDEFINED (standard) mode or in I/O MAP (flexible) mode. In the standard mode only the I/O-board position, preferably the last one, has to be selected. The input and output channels no. 9-16 will then be occupied. In the flexible mode the functions desired can be selected on any board and any channel.

4.12.1 Standard SYSTEM-I/O, inputs

There are 8 inputs which can control the execution of the user program from outside. The robot reacts on an input signal if:

- the instruction **ENABLE INTERRUPT** is executed in the user program before any of **INTERRUPT INSTRUCTION**, **INTERRUPT PROGRAM**, **JUMP TO PROGRAM** is used.
- the input signal is changed from "0" to "1" * (the input is closed)

Combined input signals must be delayed in relation to each other by at least 10 ms and activated in the correct sequence to make sure that they are processed in the correct order and interpreted correctly by the robot.

Input channel on board selected	Function
INPUT CH 9	Interrupt instruction When the signal is activated, the robot interrupts the execution of the current instruction. Execution of the next instruction in the program then begins.
INPUT CH 10	Interrupt program When the signal is activated, the robot completes the execution of the current instruction. The program execution is then stopped.
INPUT CH 9 and 10	Interrupt instruction before Interrupt program. When the signal combination is activated, the robot interrupts the execution of the current instruction in the program. Execution of the next instruction in the program is then begun. The robot completes this and program execution is then stopped. Interrupt program before Interrupt instruction. When the signal combination is activated, program execution is stopped immediately.
INPUT CH 11-15	Jump to program Each one of the inputs is connected to a fixed subprogram: <ul style="list-style-type: none"> • Input channel 11 to subprogram 1. • Input channel 12 to subprogram 2. • Input channel 13 to subprogram 3. • Input channel 14 to subprogram 4. • Input channel 15 to subprogram 5. <p>When one of these signals is activated, the robot first completes execution of the current instruction. The associated subprogram is then called and executed. Then execution continues in the program that was interrupted. Only one signal at a time may be activated.</p>

INPUT 16**Arc weld restart (AW only)**

When this signal is activated, the program execution is restarted with a correct set of arc weld data.

INPUT CH 9 and one of INPUT CH11-15**Jump to program before Interrupt instruction**

When the signal combination is activated, the associated subprogram is called immediately and executed. Then the program that was interrupted is continued by repeating the interrupted instruction.

Interrupt instruction before Jump to program

When the signal combination is activated, the robot interrupts the execution of the current instruction and execution of the next instruction in the program is completed. The associated subprogram is then called and executed. When the sub program is finished, the interrupted program is resumed at the next instruction after the last completed.

Jump to program only

The current instruction is executed before the subprogram is called. When the subprogram is finished, the interrupted program is resumed at the next instruction after the last completed.

4.12.2**Standard SYSTEM-I/O, outputs**

There are 7 outputs for indication of the status of the robot:

Output channel on board selected	Function
OUTPUT CH 9	Grip 1 Grip/release - activation of gripper no. 1
OUTPUT CH 10	Grip 2 Grip/release - activation of gripper no. 2
OUTPUT CH 11	MOTOR ON An active signal indicates that the robot is in the MOTOR ON mode, i. e. that voltage is applied to the motors.
OUTPUT CH 12	Cycle on An active signal indicates that program execution of the robot is in progress. When execution instruction by instruction, the signal is deactivated when the instruction is completed.
OUTPUT CH 13	Cycle Error An active signal indicates a malfunction in the robot during program execution. Error code is displayed.
OUTPUT CH 14	MANUAL mode An active signal indicates mode MANUAL REDUCED SPEED or MANUAL FULL SPEED.

Search stop

An active signal indicates that searching function has been stopped by a sensor signal.

4.12.3**Additional System I/O at I/O MAP****4.12.3.1****Summary of HOLD, HOLD RESET and AUTO INPUT**

Stopping of program execution followed by MOTOR OFF mode is possible by to active HOLD command. If the operating mode selector is in AUTO it is possible to resume program execution with one command by either AUTO INPUT or HOLD RESET.

Presumption: Robot in AUTO mode and a program is executing. An (EXT) HOLD stops program execution, sets HOLD-ACK and sets MOTOR OFF. If the operating mode selector remains in AUTO, it is possible to resume program execution again by AUTO INPUT (or if SYSTEM AUTO is set by (EXT) HOLD RESET).

HOLD-ACK indicates that the robot has been stopped by HOLD and has to be started by AUTO INPUT or (EXT) HOLD RESET. HOLD-ACK can be reset by MOTOR OFF. See time chart.

The following signals are involved in this functionality:

Inputs

HOLD
EXTERNAL HOLD
HOLD RESET
EXTERNAL HOLD RESET
AUTO INPUT

Outputs

HOLD-ACKnowledge

SYSTEM AUTO

Each of the signals covered in this chapter have to be defined in the I/O-MAP before they can be used. The exception is HOLD which is not always part of the SYSTEM I/O but is involved in the functionality. HOLD is inputs to the System board DSQC 256A in robots from M93.

EXTERNAL HOLD is nearly identical to HOLD.
EXTERNAL HOLD RESET is identical to HOLD RESET.

In the following text the term "(EXT) HOLD" symbolized any of HOLD and EXTERNAL HOLD. (EXT) HOLD RESET is equals to any of EXTERNAL HOLD RESET and HOLD RESET.

The (EXT) HOLD is an inverted signal i.e. it is activated when it goes from 1 to 0 and passive when it is high.

Examples of some simple ways to utilize this functionality.**A. If no signals are defined in a robot equipped with DSQC 256A:**

- A program execution can be stopped followed by a delayed MOTOR OFF in one step by HOLD on the System board. When the signals become passive it is possible to resume program execution by commencing MOTOR ON and Program Start. (If the robot doesn't have DSQC 256A, (EXT) HOLD has to be defined to get the same functionality).

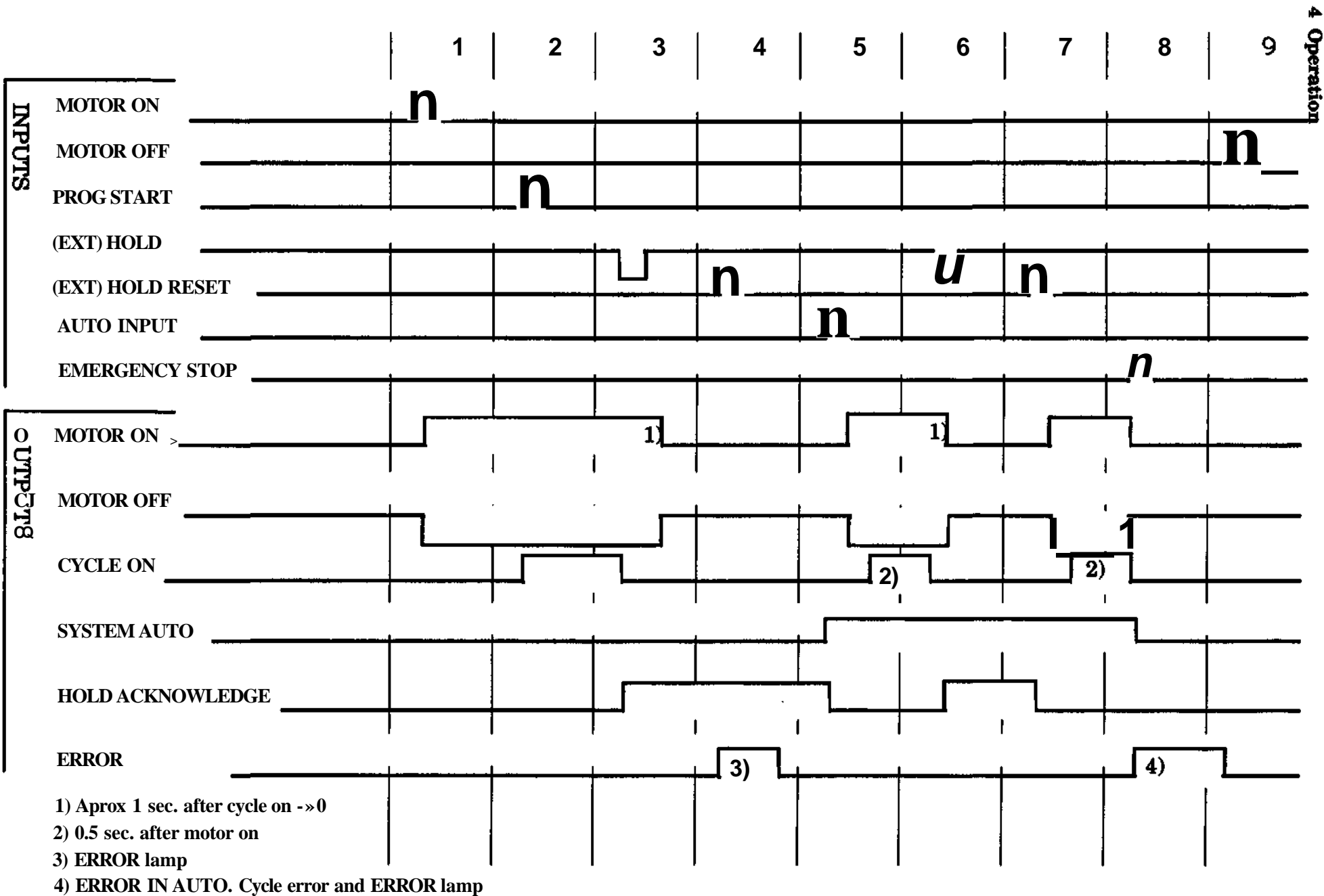
4 Operation

- B. If AUTO INPUT is defined. After commanding (EXT) HOLD, the program execution can be resumed by AUTO INPUT after that (EXT) HOLD has been set passive. AUTO INPUT carries out MOTOR ON, reset HOLD-ACK, executes Program Start and sets SYSTEM OUTPUT, if it is defined.

Note the following columns in the time chart.

Column 4 (EXT) HOLD RESET when SYSTEM AUTO is reset. (Compare column no 7.)

Column 5 AUTO INPUT starts program execution instead of (EXT) HOLD RESET.



4.12.3.2 Inputs

HOLD and EXTERNAL HOLD

HOLD (HOLD1 and HOLD2) carries out a program stop by the software. Connected in the MOTOR ON chains, the hardware on system the board delays MOTOR OFF by approx. 1.5 sec.

EXTERNAL HOLD executes both program stop and sets MOTOR OFF by software.

(If the robot is not equipped with the system board DSQC 256A, HOLD can be defined by I/O MAP and consequently only executes a program stop.)

AUTO INPUT

- 1) If program is executing, AUTO INPUT, will set the output SYSTEM AUTO.
- 2) If program execution is stopped by (EXT) HOLD, AUTO INPUT will start program execution. (HOLD ACK must be set and HOLD passive.) HOLD ACK is then reset and SYSTEM AUTO is set

Note. Neither of the outputs HOLD ACK and SYSTEM AUTO have to be defined to get this functionality.

HOLD BESET and EXTERNAL HOLD RESET

If HOLD_ACK and SYSTEM AUTO are set and (EXT) HOLD is passive, program execution is resumed by any of these two signals. This means that AUTO INPUT, which sets SYSTEM AUTO, is a "master" for (EXT) HOLD RESET. It determines when (EXT) HOLD RESET is allowed to be used.

MOV REST

The purpose of this input is to avoid collisions at a start after an abrupt stop. For detailed information see section 7.10.

4.12.4.3 Outputs

HOLD ACKNOWLEDGE (HOLD ACK)

This output is set by (EXT) HOLD, and is reset by either MOTOR OFF, AUTO INPUT or (EXT) HOLD RESET if (EXT) HOLD is passive.

HOLD ACK informs the operator or a PLC that the robot is stopped by an (EXT) HOLD signal. As long as the output is active, program execution can only be resumed by AUTO INPUT or (EXT) HOLD RESET, if (EXT) HOLD is passive.

When (EXT) HOLD becomes passive it is also possible to reset HOLD ACK by MOTOR OFF. Thus program execution has to be started by MOTOR ON and Program Start.

ERROR in AUTO

This output is set at errors stopping program execution provided that SYSTEM AUTO is set.

SYSTEM AUTO

The purpose of this output is only to inform the operator of the status for the AUTO INPUT. See above.

4.13

Remote control panel signals, Panel-I/O

The following section is valid if PANEL I/O is defined under PARAM/CHANGE/I/O-USE/PREDEF. If I/O MAP is used board, place and in- and outputs can be freely selected. A central control panel or a PLC for remote control can be used by one or more robots. The remote control panel may be used in parallel with the cabinet control panel. The inputs are called buttons and outputs are called lamps, since this is intended for a remote panel.

Input channel on board selected**Function****INPUT CH1****MOTOR ON button**

Brings the robot to MOTOR ON in AUTO mode. Voltage is applied to the robot motors.

INPUT CH2**MOTOR OFF button**

Puts the robot to MOTOR OFF. Removes voltage to the robot motors. Reset of emergency stop mode.

INPUT CH3**FROM DISK button**

Loads program from disk in AUTO mode. The disk program block 0 is loaded into the robot memory when the button is pressed twice. The loaded program block replaces the old contents in the user memory.

INPUT CH4**Disabling edit**

When set, editing of programs from programming unit is prevented.

INPUT CH5**LAMP TEST button**

Is used to check the function of the control system and remote control panel lamps. All lamps are lit when the button is pressed. (Except Robot/Ext, axes and 1-2 key functions.

INPUT CH6**PROG STOP button**

Interrupts program execution and synchronization procedure.

INPUT CH7**PROG START button**

Starts program execution in AUTO mode. In special circumstances the AUTO button on the programming unit has to be pressed before starting the program execution.

INPUT CH8**SYNCHRONIZATION button**

This button is used to restart the robot after a power failure during program execution. The synchronization of external axes is started when the button is pressed.

4.13

Remote control panel signals, Panel-I/O

The following section is valid if PANEL I/O is defined under PARAM/CHANGE/I/O-USE/PREDEF. If I/O MAP is used board, place and in- and outputs can be freely selected. A central control panel or a PLC for remote control can be used by one or more robots. The remote control panel may be used in parallel with the cabinet control panel. The inputs are called buttons and outputs are called lamps, since this is intended for a remote panel.

Input channel on board selected**Function**

INPUT CH1

MOTOR ON button

Brings the robot to MOTOR ON in AUTO mode. Voltage is applied to the robot motors.

INPUT CH2

MOTOR OFF button

Puts the robot to MOTOR OFF. Removes voltage to the robot motors. Reset of emergency stop mode.

INPUT CH 3

FROM DISK button

Loads program from disk in AUTO mode. The disk program block 0 is loaded into the robot memory when the button is pressed twice. The loaded program block replaces the old contents in the user memory.

INPUT CH4

Disabling edit

When set, editing of programs from programming unit is prevented.

INPUT CH5

LAMP TEST button

Is used to check the function of the control system and remote control panel lamps. All lamps are lit when the button is pressed. (Except Robot/Ext, axes and 1 - 2 key functions.

INPUT CH 6

PROG STOP button

Interrupts program execution and synchronization procedure.

INPUT CH 7

PROG START button

Starts program execution in AUTO mode. In special circumstances the AUTO button on the programming unit has to be pressed before starting the program execution.

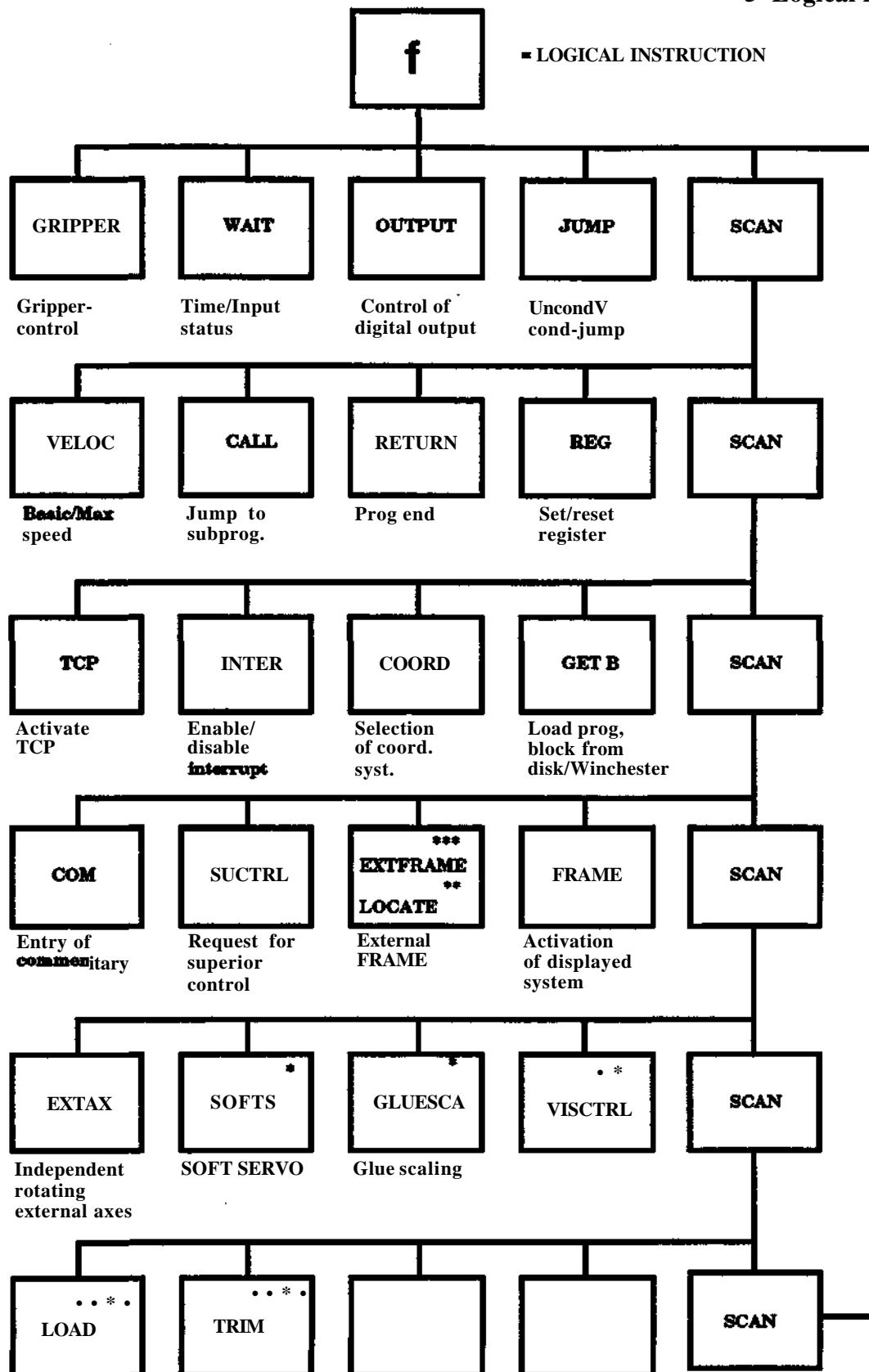
INPUT CH 8

SYNCHRONIZATION button

This button is used to restart the robot after a power failure during program execution. The synchronization of external axes is started when the button is pressed.

5. Logical instructions menu

Section	Page
5.1 GRIPPER	5:5
5.2 WAIT	5:7
5.2.1 WAIT (TIME)	
5.2.2 WAIT (INPUT)	
5.3 OUTPUT	5:9
5.4 JUMP	5:9
5.4.1 JUMP (unconditional)	
5.4.2 JUMP (conditional)	
5.4.3 JUMP (SEARCH)	
5.4.4 JUMP (NINPOS)	
5.5 VELOC	5:12
5.6 CALL	5:13
5.7 RETURN	5:15
5.8 REG	5:15
5.8.1 REG (FETCH)	
5.8.2 REG (SET)	
5.8.3 REG (TRANSFER)	
5.8.4 REG (LOC)	
5.9 TCP	5:18
5.10 INTER	5:19
5.11 COORD	5:21
5.12 GET B/ADD BLOCK	5:21
5.13 COM	5:22
5.14 SUCTRL	5:23
5.15 FRAME/FRAME DEFINE	5:24
5.16 SOFTS	5:25
5.17 EXTAX	5:26
5.18 EXTFRAME	5:27
5.19 GLUE SCA	5:27
5.20 LOAD (IRB 6000 only)	5:27
5.21 TRIM (IRB 6000 only)	5:27
5.22 LOCATE and VISCTRL	5:28



•MH/GL/SW
 •* If Vision is defined
 •• AW only
 ••• IRB 6000 only



5.1 GRIPPER

Gripper function

Menu: INSTRUCTION

Function: GRIPPER

Means:

A gripper, a welding gun or corresponding tool is operated during program execution.

Facts:

The gripper can assume two positions, GRIP or RELEASE. With gripper operations the robot must remain still until the movement is completed. For this reason, a waiting time is given in the instruction.

The gripper instruction is implemented in two versions; one for MH/GL/SW robots and one for AW robots.

In the MH/GL/SW case, up to eight tools can be controlled and the release/grip movement for the chosen gripper is explicitly programmed in the instruction. Only one gripper can be controlled by one instruction. In this case the actual positions of the grippers does not matter when programming. The connection of the grippers is described briefly below (see Outputs for gripper) and in detail in the Installation S3.

In the AW case, only grippers can be controlled. The release/grip function is implicitly defined by the actual positions of the grippers when programming, i.e. the grippers must be positioned manually before the instruction is programmed. Both of the grippers can be controlled in one instruction.

Executed:

During program execution, the gripper is positioned to the position decided when programming. The robot then stands still until the waiting time has expired.

Procedure:

Instruction for gripper operation MH/GL/SW

Menu: INSTRUCTION

Function: GRIPPER

1. Select GRIPPER under the INSTRUCTION menu.
2. Specify gripper number 1-8.
The gripper number last used will be taken if ENTER is pressed without giving any gripper number.
3. Select GRIP or RELEASE.
4. Specify if the waiting time for gripper operation is to be entered with the numerical button set (YES) or have its pre-programmed value 0.3 s (NO).

If NO is selected, the instruction is now completed. Otherwise continue with point 5 below.

5. Specify the waiting time in the range 0.1 - 9.9 in seconds with one decimal.
The instruction is thereby completed.

Manual control of gripper MH/GL/SW

Menu:  RELEASE or  GRIP

Manual operation of the "GRIPPER" via the programming unit is done as shown below:

The function is divided in two cases:

- Two grippers defined (default value)
The button for gripper 1 (grip, release) handles gripper one
The button for gripper 2 (grip, release) handles gripper two
- More than two grippers defined (n pcs)
The button for gripper 1 (grip, release) handles gripper one
The button for gripper 2 (grip, release) gives the question which gripper you want to control. Valid answers range from 2-8.
The last gripper number is used as a default value, and only ENTER is needed.



**The gripper is usually opened directly when the gripper button is pressed.
Make sure no detail is dropped causing damage or injury.**

Procedure:

Instruction for gripper operation (AW)

- 1 Set the gripper or the welding gun in the position required with gripper buttons on the operators panel.
- 2 Select GRIPPER under the INSTRUCTION menu.
- 3 Specify if the waiting time for gripper operation is to be entered with the numerical button set (YES) or have its pre-programmed value 0.5 s (NO).

If NO is selected, the instruction is now completed. Otherwise continue with point 4 below.
- 4 Specify the waiting time in the range 0.1 - 9.9 in seconds with one decimal.

The instruction is thereby completed.

Outputs for gripper (MH/GL/SW)

It is possible to connect grippers (in a multi-gripper) to the robot via digital outputs.

The robot will automatically reserve a group of logical outputs. The reservation will only affect the number of logical outputs necessary for the specified number of grippers

Example: Five grippers are to be connected to the robot system.

Gripper 1 and 2 are connected to their respective outputs in system I/O.

The following gripper data is entered in the system parameter:

1. Five grippers
2. Logical output 14 (output for gripper 3)

When connecting the grippers 3-5:

- Gripper 3 is connected to the logical output 14
- Gripper 4 is connected to the logical output 15
- Gripper 5 is connected to the logical output 16

The logical outputs 17-19 are not reserved for gripper 6-8, because five grippers has been specified under 1 above.

5.2 WAIT

Pause in program running

Menu: INSTRUCTION

Function: WAIT

Means:

The running of the program is interrupted:

- A predetermined time.
- Until 1-8 digital inputs have assumed the required status.
- Until 1-8 digital inputs have assumed the required status, or the supervision time has expired.

Facts:

A waiting time can be determined in seconds with two decimals in the range 0 - 99.99 s.

The correct status before restart of program execution at each of the inputs included in the condition can be 1 or 0.

If required, a maximum permitted time before restart can be set in complete seconds in the range 0 - 320 s. A time of 60 s is preprogrammed. See Installation S3.

Used:

When program execution must be interrupted at a particular position in the program while another machine, or the operator, performs some other operation.

Executed:

During program execution, the program remains inactive:

- Until the time selected has expired.
- Until all 1 - 8 inputs in the condition have assumed their correct status or, if supervision is selected, the time has expired.

If CLINK is defined, a WAIT-instruction should be used if there are more than 15 instructions between two POS-instructions, two WAIT-instructions or one POS and one WAIT-instruction. Then the delay time should be at least 0.01 seconds.

5.2.1

WAIT (TIME)

Procedure:

Instruction with waiting time

1. Select WATT under the INSTRUCTION menu.
2. Select TIME.
3. Specify the time 0.01 - 99.99 s.

The instruction is now complete.

5.2.2

WAIT (INPUT)

Instruction with input conditions

1. Select WAIT under the INSTRUCTION menu.
2. Select INPUT.
3. State the number of the first input in the condition.
4. Give the correct status for the first input.
5. Repeat the points 3-4 above for a new input (YES) or go to point 6 (NO).
6. Specify (with YES) if there is to be supervision of the maximum permitted waiting time before restart of program execution. Continue in this case to point 7.

The instruction is otherwise now complete.

7. Specify the time 0 - 320 seconds. Press ENTER directly if the preprogrammed value (normally 60 s) is to apply.

The instruction is now complete.

A program example is shown below:

```
160 POS V=100 %
170 WATT UNTIL INP 6 =1 MAX = 60 S
180 JUMP TO 200 IF INP 6 = 1
190 CALL PROG 1000
200 POS V=100 %
```

5.3 OUTPUT

Change of the status of an output during program execution

Menu: INSTRUCTION

Function: OUTPUT

1. Select OUTPUT under the INSTRUCTION menu.
2. State the number of the output concerned.
3. Select one of the following alternatives:
 - Set the output to 1 irrespective of its current status.
 - Set the output to 0 irrespective of its current status.
 - Change the status.
 - Send a 200 ms pulse (negative if the output is a 1, positive if the output is at 0 when the command is executed).

The instruction is now completed.

5.4 JUMP

Jump within the program

Menu: INSTRUCTION

Function: JUMP

Means:

A program execution can jump to another instruction unconditionally or when one or more conditions are fulfilled.

Facts:

Jumps can be made both forwards and backwards in the program

- * Unconditionally.
- Under 1-8 conditions each of which can be one of the following:
 - One digital input is to be 1 or 0.
 - One digital output is to be 1 or 0.
 - The value in a number register 0 - 99 is to be either
 - greater than (>)
 - less then (<)
 - equal to (=)
 - or
 - not equal to < - > a reference value. (<>)

Used:

Generally, when program execution is to be controlled by internal or external conditions.

Executed:

After the jump instruction has been executed, a jump will be performed to the instruction with the number given in the jump instruction. If the jump instruction is conditional, the jump will be performed only if the condition is fulfilled, otherwise the execution will go on with the instruction following the jump instruction.

5.4.1

JUMP (unconditional)

Executed:

An unconditional jump is performed as followed:

1. Select JUMP under the INSTRUCTION menu.
2. Specify which instruction in the program is to be executed instead of that immediately following.
3. Specify that there are no conditions for the jump with NO.

The instruction is complete.

5.4.2

JUMP (conditional)

Procedure:

Jump instruction with 1-8 conditions

1. Select JUMP under the INSTRUCTION menu.
2. Specify which instruction in the program is to be executed instead of that immediately following when all conditions are satisfied.
3. Specify that conditions for the jump is to be set with YES.
4. Specify if the condition is to apply to either:
 - A register, with REG.
 - An input, with INPUT.
 - An output, with OUTPUT.
 - Search condition

Continue through the conditions selected (see "Facts", above) until the question CONTINUE? is presented on the display.

5. Specify if several conditions are to be set for the jump (YES). Repeat in this case points 3 and 4 for the next conditions.

If NO, the instruction is now concluded.

5.4.3 JUMP (SEARCH)

Jump with search stop conditions

Search stop can be used as a jump condition. A jump can be programmed to take place when the search stop is either active (one) or inactive (0). Through these the following program execution can be controlled by the search result. See the example below.

Example: Main program

```
80 POS SEARCH
90 JUMP TO 110 IF
  SEARCH = 0
100 CALL PROG 5
110 POS
```

In the program above, the robot searches for an object when instruction so is executed. If the object is encountered SEARCH becomes = 1 and subprogram 5 is called. If however no object is encountered during the search, SEARCH remains = 0 and the program call in the instruction 100 is omitted.

Further information about jump in program, see "Jump within the program".

If the robot contains extra digital outputs, the output SEARCH STOP can control peripheral equipment. See Installation S3.

The instruction is now programmed and no further arguments can be added.

4. Press ENTER
5. Press YES
6. Press SEARCH
7. If the search stop is to be a jump condition press =1 otherwise press =0.

5.4.4 JUMP (N INPOS)

Jump if not in position

Means:

A conditional jump instruction which causes a jump in the program if the robot does not reach the programmed position. No other conditions are permitted together with N INPOS.

Facts:

To use the function, soft servo must be and activated. See section 9.5.

The allowed position error can be chosen by specifying different zones. When using this instruction the position of the robot must not be programmed with a FINE zone.

5 Logical instruction menu

Program example:

100	SOFTSERVO 1	Activate soft servo
110	POS V = 100 % PATH	Position
130	JUMP TO 300 IF N INPOS	Jump if the position is incorrect
	FINE	
140	SOFT SERVO 0	Deactivate soft servo
150...		Assembly is correct. Continue.
300	POS V = 100 % PATH	Assembly is not correct. Move the tool to free space and make error handling.
310	SOFT SERVO 0	
320...		

Used:

The instruction can be used to check if the robot has reached the desired position or not. If the position is not reached, a jump is made in the program and corrections of the error can be done.

Executed:

As the normal conditional JUMP instruction.

Procedure:

Menu: INSTRUCTION

Function: JUMP

1. Select JUMP under the INSTRUCTION menu.
2. Specify which instruction in the program is to be executed instead of that immediately following when the condition is satisfied.
3. Specify that conditions for the jump are to be set with YES.
4. Select N INPOS.
5. Specify zone by selecting CORNER1, CORNER2 or PATH.

The instruction is now completed.

5.5 VELOC

Change of basic and maximum speeds

Menu: INSTRUCTION

Function: VELOC

Means:

Definition of basic and maximum speeds for a number of instructions.

Facts:

The speed with which a movement is to be performed to a programmed position is built by three different components:

- An absolute basic speed with pre-programmed value 1000 mm/s.
- A relative percentages speed specified for each individual instruction as a percentage of the basic speed with pre-programmed value 0 %.
- An absolute max. speed with pre-programmed value 2500 mm/s.

5 Logical instruction menu

The relationship between these is shown in the following example:

110 V = 1000 MM/S MAX = 2000 MM/S

120 POS V = 100 %

130 V = 1000 MM/S MAX = 200 MM/S

140 POS V = 25 %

Basic and max. speeds are specified for the succeeding instructions.

Movement with 100 % x 1000 mm/s = 1000 mm/s

A new maximum speed is specified for the succeeding instructions.

Motion with 200 mm/s. The maximum speed is less than the programmed speed.

Used:

When basic and maximum speeds are to be redefined (from their pre-programmed values).

Performed:

After definition of the speeds during program execution these apply until a new definition is encountered in some program, irrespective of the number of program changes which have occurred before.

Procedures:

1. Select VELCO under INSTRUCTION menu
2. Specify both basic and maximum speeds, even if only one of these is to be redefined.

 The speed of the external axes and robot may, when running in robot coordinates, exceed the programmed speed.

5.6 CALL

Program exchange

Menu: INSTRUCTION

Function: CALL

Means:

The program execution continues in a subprogram, performed completely or module by module before execution of the program interrupted is continued.

Facts:

Program change is performed via call. The program called can be performed when called an optional number of times from 1 - 99 before return to the calling program is executed.

The number of the program to be called can be specified in the instruction, either:

- Directly.
- Indirectly via the value in a numerical register 0 - 119.

Used:

Generally, when program exchange is necessary during program execution.

Executed:

1. The call instruction is issued.
2. The execution of the calling program is interrupted.
3. The program called (or the module within the program) is executed 1-99 times.
4. The program called continues with the instruction after the call instruction.

Procedure:

1. Select CALL under the INSTRUCTION menu.
2. Specify if the number of the programmed called is to be selected via a register (YES) or directly (NO).
3. Specify the program number or the register number depending on the selection in point 2 above (0 - 9999 or 0 -119).
4. Specify if the program called is a pattern program (YES) or not (NO).

If YES, continue with point 5 below. If NO, go directly to point 6.

5. Specify the number register 0-119 which is to control the call of the modules within the program.
6. Specify if the program called is to be performed 2-99 times (YES) or once only (NO) at the time of the call.

When the repetition is to be performed, continue to point 7. The instruction is otherwise now complete.

7. Specify how many times the program is to be performed (2 -119).

The instruction is now completed.

5.7 RETURN

End of instruction sequence

Menu: INSTRUCTION

Function: RETURN

Means:

The last instruction in a program.

In a pattern program it is also the last instruction within a program module, see section 11.4.

Facts:

The instruction returns the program running to:

- The first instruction in the same program.
- The calling program where the running continues with the first instruction after the call when the program called is executed the number of times required.

Procedure:

Instruction for program end (or end of program module)

1. Select RETURN under the INSTRUCTION menu.

The instruction is then completed.

5.8 REG

Arithmetic handling

The values stored in the registers can be handled with arithmetics, i.e. be added, subtracted, multiplied or divided, with each other or with constant values. The values can be positive as well as negative.

Example:

$R1 = R2 + 5$ ($R1$, $R2$ and $R3$ are data registers).
 $R2 = R3 + R1$
 $R1 = R3 + 1$

Division with 0 or overflow causes PROGRAM RUN ERROR 27.

Division is performed without respect to decimals, i.e. $8/5 = 2.7/1 = 1$, and the decimal part is adjusted. If the decimal part is 0.5 or greater it is adjusted upwards to the next unit value.

Range: -32 768 to 32 767

5.8.1 REG (FETCH)

Accessing of register value

A robot register is given a value from an external source.

Menu: INSTRUCTION

Function: REG

- 1 Select REG under the INSTRUCTION menu.
- 2 Select FETCH.
- 3 Specify register number 0 - 119.
4. Specify:
 - Digital port 11-14
 - Analog port 31-34

The instruction is hereby concluded.

	Port	Input
Digital	11	1-4
	12	13-16
	13	17-24
	14	25-32
Analog	31	1
	32	2
	33	3
	34	4

See Installation manual.

Digital ports 11 and 12 can access four bits, i.e values between 0-15, ports 13-14 can access eight bits, i.e values between 0-255. The analog ports have a scale factor of 10 mV and 0.02 mA respectively corresponding to value 1 in the register.

5.8.2 REG (LET)

Procedures:

Arithmetic handling of register values

Menu: INSTRUCTION

Function: REG

1. Select REG under the INSTRUCTION menu.
2. Select LET.
3. Specify register number.
4. Select VALUE, REG eller REG- depending on if a constant value, a positive or a negative value is to be entered.
5. Define the register number or the constant value.
6. Define whether the earlier value is to be added, subtracted, multiplied or divided with the following specified values.

5 Logical instruction menu

7. Define VALUE or REG depending on if a constant value or a register number is to be entered.

8. Define the register number or the constant value.

9. Check the final formula on the upper line of the programming unit's display.

Finally, leave the menu by, for example, pressing MANUAL

5.8.3

REG (TRANSFER)

Transmission of register value during program execution

A robot register value is transmitted to an external.

Menu: INSTRUCTION

Function: REG

1. Select REG under the INSTRUCTION menu.
2. Select TRANSFER.
3. Specify register number 0-119.
4. State port number in accordance with the following:
 - Digital port 1 - 4
 - Analog port 21-24

	Port	Input
Digital	1	1-4
	2	13-16
	3	17-24
	4	25-32
Analog	21	1
	22	2
	23	3
	24	4

See Installation manual.

The instruction is hereby concluded.

Digital ports 1 and 2 can transmit values of 0-15 units, ports 3 and 4 can transmit 0-255 units. The analog ports have a scale factor of 10 mV = one register unit and 0,02 mA = one register unit.

5.8.4

REG (LOC)

Robot positions

The different positions of the robot can be stored in the data registers (REG NO 0-119). The three coordinate values (x, y and z) are stored in three consecutive registers. With the help of register arithmetics (see above), displaced positions can be calculated for example.

The stored position is the location for the active TCP when the instruction is executed. The

5 Logical instruction menu

location is expressed with a resolution of 0.1 mm or 0.01 inches and is defined relative the object coordinate system.

A specific position in space will cause different values depending on the FRAME or REFPOINT ACTIVE. This means that when the displacement between the two points is to be calculated, the same FRAME (REFPOINT) must be active for both points during storage in registers.

During entry to registers, the first register number only is specified and the two remaining values will be stored in the two consecutively numbered registers.

Entry of robot position

1. Select REG under the INSTRUCTION menu.
2. Select LOC.
3. Specify register number 0-119.

The instruction is hereby complete.

5.9 TCP

Activation of TCP during program execution

Menu: INSTRUCTION

Function: TCP

Previous definition of the TCP-selected is assumed

1. Select TCP under the INSTRUCTION menu.
2. Specify TCP number 0-19 for ordinary TCP. MH/GL/SW only: 20 - 29 are reserved for room fixed TCP.

Note!

When the instruction is programmed the TCP selected is also activated.



Activation of an incorrect TCP will cause the robot to perform unintended motions.

When the TCP instruction is executed during program execution, the internal representation of the TCP position will change but the TCP instruction itself will cause any robot motion. The first position instruction after the TCP instruction, however, will move the robot using the new TCP to the programmed position.

Please note, that if a path is programmed with zones, no cornerpath will be generated after the TCP instruction but the robot will move directly to the following position with the new TCP activated. See section 3.6. To avoid problems, the TCP instruction should be executed after a POS FINE instruction, i.e. in a well defined position.

Warning text for TCP

In certain circumstances when editing and interchanging programs, there is an increased risk that an incorrect TCP is active. A warning text is presented on the programming unit to indicate that an incorrect TCP may be active.

The warning text is obtained with execution of the program after the following operations if a TCP instruction has not been executed before the first POS instruction.

- * Exchange of active instruction during editing.
- * Exchange of active TCP during manual operation.
- * Exchange of active program during editing.
- * Exchange of active program during system initialization.
- * Exchange of active program during CLEAR.

5 Logical instruction menu

- Exchange of active program during downloading from a disk.
- Exchange of active program during synchronization.

The warning text on the programming unit has the following appearance:
TCP xx IS ACTIVE!

An error code as follows is presented in the error buffer and on the monitor:

504 PROGRAM RUN ERROR 72

The TCP must be checked when the warning is presented. Continue execution by pressing again program start or instruction start.
For Room Fixed TCP see chapter 3.3.2.

5.10 INTER

Interrupt in the program execution via direct-acting inputs

Menu: INSTRUCTION

Function: INTER

Means:

Program execution can be interrupted with the help of digital signals from peripheral equipment via direct-acting inputs. These interrupts can be controlled to affect or not affect the program running during different stages of the work cycle.

Facts:

Interrupt can be performed in one of the following ways:

- Current instruction interrupted and the next instruction is begun.
- Program stop when the current instruction is performed.
- Immediate program stop.
- Jump to one of the subprograms 1-5 when the current instruction is executed.
- Immediate jump to one of the subprograms 1-5.

A small program example is shown below:

60 DISABLE INTERRUPT

Interrupt caused by direct-acting inputs are not permitted to affect program execution from and including instruction number 70.

Assume that input JUMP TO PROGRAM 5 is connected to the peripheral equipment. If the input is then set to 1, the program execution is not affected as the interrupt request is not registered by the robot.

250 ENABLE INTERRUPT

From this instruction, interrupt in program execution is permitted from direct-acting inputs.

When the signal is set to 1 before instruction 250 is entered, the robot is again unaffected by the signal because the signal must be set to 1 after instruction 250 is executed.

The following happens however when JUMP TO PROGRAM 5 is set to 1 after instruction 250 has been executed:

1. The instruction under execution, e.g. instruction 340, is completed.
2. Subprogram 5 is called and executed.

5 Logical instruction menu

3. Program execution returns to the program interrupted, i.e. execution continues with instruction 350.

An instruction in the program which enables or disables interruptions in the execution applies until a new instruction of the same type is encountered.

The direct-acting functions according to the above are connected, as hardware, during the installation of the robot. See Installation S3.

Used:

When the peripheral equipment may enable the program execution during the work cycle. The instruction determines when this is allowed.

Executed:

If interruption of program execution is currently allowed, the robot immediately reacts to signals **from** the direct-acting inputs. Note that there are certain limitations which are described in the Installation S3.

Procedures:

1. Select INTER under the INSTRUCTION menu.
2. State whether interruptions are to be allowed, ENABLE, or not allowed, DISABLE, for the following instructions.

5.11 COORD

Change of coordinate system for programmed execution

Menu: INSTRUCTION

Function: COORD

1. Select COORD under the INSTRUCTION menu.
2. Select RECT(angular) MODRECT or ROBOT.

5.12 GETS

An instruction for program loading in a robot program is prepared as follows:

Programmed loading of program block

Menu: INSTRUCTION

Function: GET B

1. Select GET B in the menu INSTRUCTION.
2. Specify whether the block number is to be selected via registers (YES) or directly (NO).
3. Specify the block number or the register number depending on the selection made in point 2 above (0-9999 or 0-119).
4. Select either of:
 - FR DISK The program block is loaded from the diskette.
 - FR WIN The program is loaded from Winchester memory (if such is connected)
5. Select either of:
 - REPLACE The contents of the program memory are replaced by the block to be loaded. The instruction GET BLOCK is completed when REPLACE is pressed.
 - ADD ALL The contents of the user memory are supplemented by the block to be loaded. However, any duplicates in the program memory are replaced by the programs loaded.
6. When ADD ALL has been selected, a question is displayed whether any program in the program memory is to be erased before the loading is executed. Press ENTER if erasure is not required. The instruction ADD BLOCK is completed.
7. Specify the number of a single program to be erased and press ENTER if a program is to be erased. If a series of programs are to be erased, specify the number of the first and last programs in the series and press ENTER. The instruction ADD BLOCK.. ERASE... is now completed.

- Note!**
- When a GET BLOCK instruction has been executed, program execution continues from the first instruction in the main program of the newly loaded program block.
 - When an ADD BLOCK instruction has been executed, program execution continues from the next instruction in the program which called for program loading. ADD BLOCK can be executed from main program level only, not from a subprogram.
 - Program blocks containing the same program number as the program requesting the loading, can not be loaded. An error message will be displayed and program execution is stopped.
 - Avoid program printout and displacement of positions when mass storage is used.

5.13 COM

Program stop with comments

Menu: INSTRUCTION

Function: COM

Means:

Comments can be entered in the program using the programming unit. When the optional function "Program Printout" is provided, a comment can be inserted in the program from the printer.

Facts:

The comments in the program, with associated program stop can consist of up to 72 ASCII code characters, including spaces, read in from a keyboard or 24 characters read in from the programming unit.

Used:

The commentary in the program with associated program stop is used in program sequences in which it is necessary to be able, simultaneously, to:

- Stop program execution.
- Receive a readout on a monitor or on the programming unit, or a printout on a printer indicating why the robot is at standstill. Even without program stop, the comment takes time to present. Comments may therefore not be suitable in all situations.

Executed:

When a commentary instruction with associated program stop is performed, the robot stops and the commentary is presented. If the program stop is not used, comments are presented on the display and monitor when the instruction with comment is executed, and remains until the next comment is presented.

Procedure:

Instruction with comments and associated program stop (from keyboard)

1. Select COM under the INSTRUCTION menu.
2. Select YES on the question "FROM TERMINAL?" if an external keyboard is to be used.
3. Specify whether program stop is to be associated with the commentary, or not.
4. Enter the comments from the keyboard, with up to 72 characters including spaces.
5. Make a carriage return on the keyboard.



Instruction with comments and associated program stop (from programming unit)

1. Select COM under the INSTRUCTION menu.
2. Select NO on the question "FROM TERMINAL?" if the programming unit is to be used.
3. Specify whether program stop is to be associated with the commentary, or not.
4. Enter the comments from the programming unit, according to the figure below.
5. Enter "." when the comment is completed. The text string is otherwise concluded automatically after 24 characters "-" for erasure.

Entry is performed with the help of the multi-function buttons and numbers. To enter, for example, a P, first press the button F3 and then "4".

* # is available but does not appear.

5.14 SUCTRL

SUCTRL, request for superior computer control

The function SUCTRL is used when a programmed request for superior computer control is to be sent to the SC. When the instruction is executed during programmed execution, the robot system concerned will send a spontaneous status message to the S.C. The robot is then prepared to react immediately on order from the S.C.

If stop is programmed, a stop-command must be given from the S.C. before a start of program execution can be requested again. When the robot is in the instruction SUCTRL before a STOP command is given from the S.C. the programming unit is blocked. It is possible to execute commands from the S.C.

If required, the request for superior control can be cancelled manually by means of a program stop via the programming unit. In addition a request for superior control can also be cancelled by a program stop via an digital input.

Program the function in accordance with the following:

1. Press the function control button INSTR
2. Press the function button SCAN three times.
3. Press the function button SUCTRL.
4. If the robot is to continue to execute without halting, press YES. If it is to stop and await a STOP command from the S.C, press NO.
5. If a register value is to be sent to the S.C, press REG NO and enter the required register number (0-119)

The instruction SUCTRL is now completely programmed and no further arguments can be added.

5.15

FRAME/FRAME DEFINE

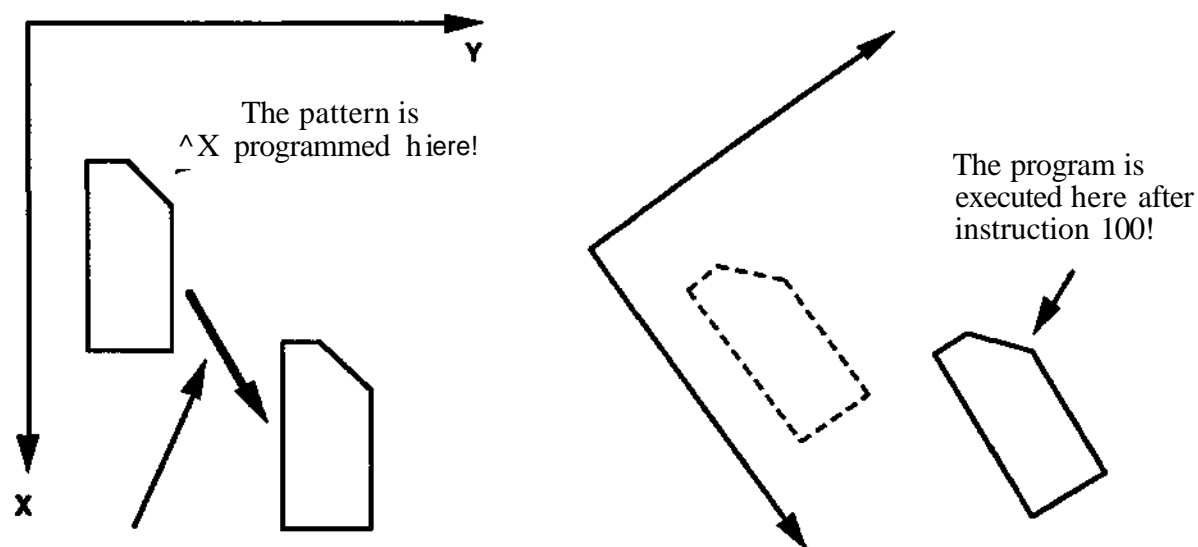
Activation of program displacement

Function: FRAME

Menu: INSTRUCTION

1. Select FRAME under the INSTRUCTION menu.
2. Specify 'which of the program displacements 1 - 5 is to be activated.
3. Select SET.

Note that the frame displacement and a reference point displacement can be active simultaneously. In such a case, the frame displacement is always calculated first, and consequently, the reference point displacement will be calculated and superpositioned upon the frame displacement, according to the figure below.



This displacement
is activated in
INSTRUCTION 50

```

10
"
"
"50 REFP
"
"
100 FRAME
"
"
"
"
```

See section 3.9.2 for a more detailed description of the FRAME function.

See section 6.14 for programmed activation of desired FRAME.

See section 2.6 for manual definition of FRAME.

See section 10.7 for automatic definition of FRAME.

5.16

SOFTS (MH/GL/SW only)**Instruction programming**

Any set of defined softness or normal speed control, can be activated manually or during program execution. An attempt to execute a SOFT SERVO instruction with an undefined number will cause the program execution to stop and display an error message (C'504 PROGRAM RUN ERROR 70").

A SOFT SERVO instruction is programmed as follows:

1. Select SOFTS under the INSTRUCTION menu.
2. Specify desired set of soft servo. Number 0 provides normal speed control on all axes.

Position instructions executed while SOFT SERVO is activated should not be FINE.

All nine sets of softness percentages form a part of the system parameters, possible to store on a disk for later use.

for detailed information, see section 3.8.

See section 9.5 for definition of softness percentage.



When using the function Soft Servo, the robot speed and path are changed-

Note. In order to prevent abrupt motions when activating and deactivating SOFT SERVO, a WAIT instruction can be programmed before and after the SOFT SERVO sequence.

5.17

EXTAX Independent External Axes

EXTAX is a function which allows the external axes of a robot to be moved independently of the normal positioning. Independent mode or normal mode may be selected dynamically during the robot program execution by using instructions "AXIS x START" or "AXIS x STOP" respectively.

During independent mode, an axis may be rotated in one direction for long periods, and because of this, it is necessary to allow it to have an undefined working area, so that it does not become limited. The working area can be selected as undefined or defined during definition of the system parameters (See Installation S3). As axes may have a defined or undefined working area in both independent and normal movement modes, there is no restriction.

In some cases, an external axis may be a rotational device (e.g a rotating table), in some cases it may be linear device (e.g. track motion). As a consequence of this, it is possible to specify the axis type as rotational or linear during definition of the system parameters.

When EXTAX is used, the parameter ROTAX should be set to 1. This means that, for example, a rotating table will only ever rotate up to one revolution to reach a position. For such a device it is important to define the external axis gear ratios correctly.

Programming of external axes to start independent movement will be done by the following programming unit interaction:

Menu: INSTRUCTION

Function: EXTAX

1. Select EXTAX under the INSTRUCTION menu.
2. Specify the number of the axis in question.

The valid range of "AXIS NO=" is 7-12.

There will be NO check to ensure that the programmed axis is defined because it is possible to modify the parameters and undefine an axis at some later time. Instead, operations on undefined axes will be ignored at execution time as is already the case with normal external axes programming.

3. Select START.
4. Specify the velocity of the external axis in question (RPM if rotational, UNITS/S if linear).

The valid range for "VELOCITY (RPM)=" is -5000...+5000 rpm.

There will be NO check to ensure that the programmed speed does not exceed the maximum speed because it is possible to modify the parameters and change the maximum speed at some later time. Instead, the speed will be limited at execution time if necessary.

Programming of external axes to stop independent movement and return to normal movement mode will be done by the following programming unit interaction:

Menu: INSTRUCTION

Function: EXTAX

Step 1 and 2 are the same as for the start instruction above.

3. Select STOP.

5.18**EXTFRAM (AW only)**

The instruction makes it possible to program interpolated movement between the robot and one external axis.

See section 12.6.5 for detailed description and how to program.

5.19**GLUE SCA (MH/GL/SW only)**

Used for gluing only. See chapter 13.

5.20**LOAD (1KB 6000 only)**

The instruction makes it possible to program different wrist loads. Thus the performance of the robot can be optimized to the actual load in order to reduce cycle time. See 11.8 for detailed information. 20 different loads can be programmed.

1. Selected LOAD under the instruction menu.
2. Specify LOAD number 0-19.

LOAD 0 is defined as a system parameter, see Installation S3.
LOAD 1-19 is defined according to section 9.5.

The defined LOAD 0 values are stored as system parameters and LOAD 1-19 values as user parameters like TCP.

5.21**TRIM (1KB 6000 only)**

To be able to optimize the robot's movement into FINE points in a movement, a function TRIM is available. The TRIM function makes it possible to adjust the movement between two FINE points, according to mod. A described in Chapter 4, in the optimum manner. As well as mode A, it is only available for 1KB 6000.

The function is programmed as an ordinary instruction and can change the servo position control with ± 100 scale divisions.

Preselected value is 0, the same as when disconnected. The TRIM factor is a part of the robot program and will be saved on a disk or superior computer together with the robot program.

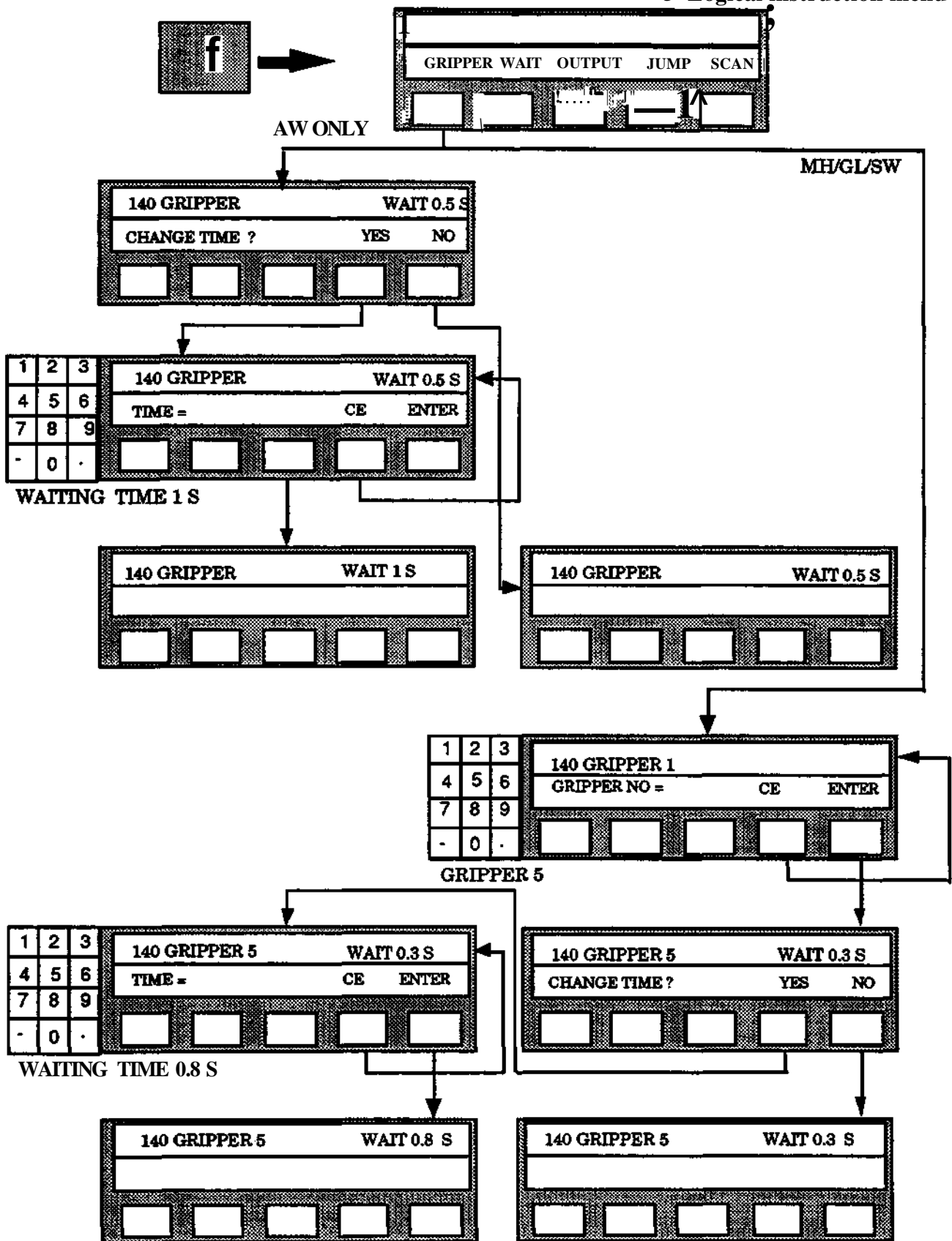
All fine points programmed according to a TRIM instruction will be using the programmed factor until a new TRIM instruction is programmed.

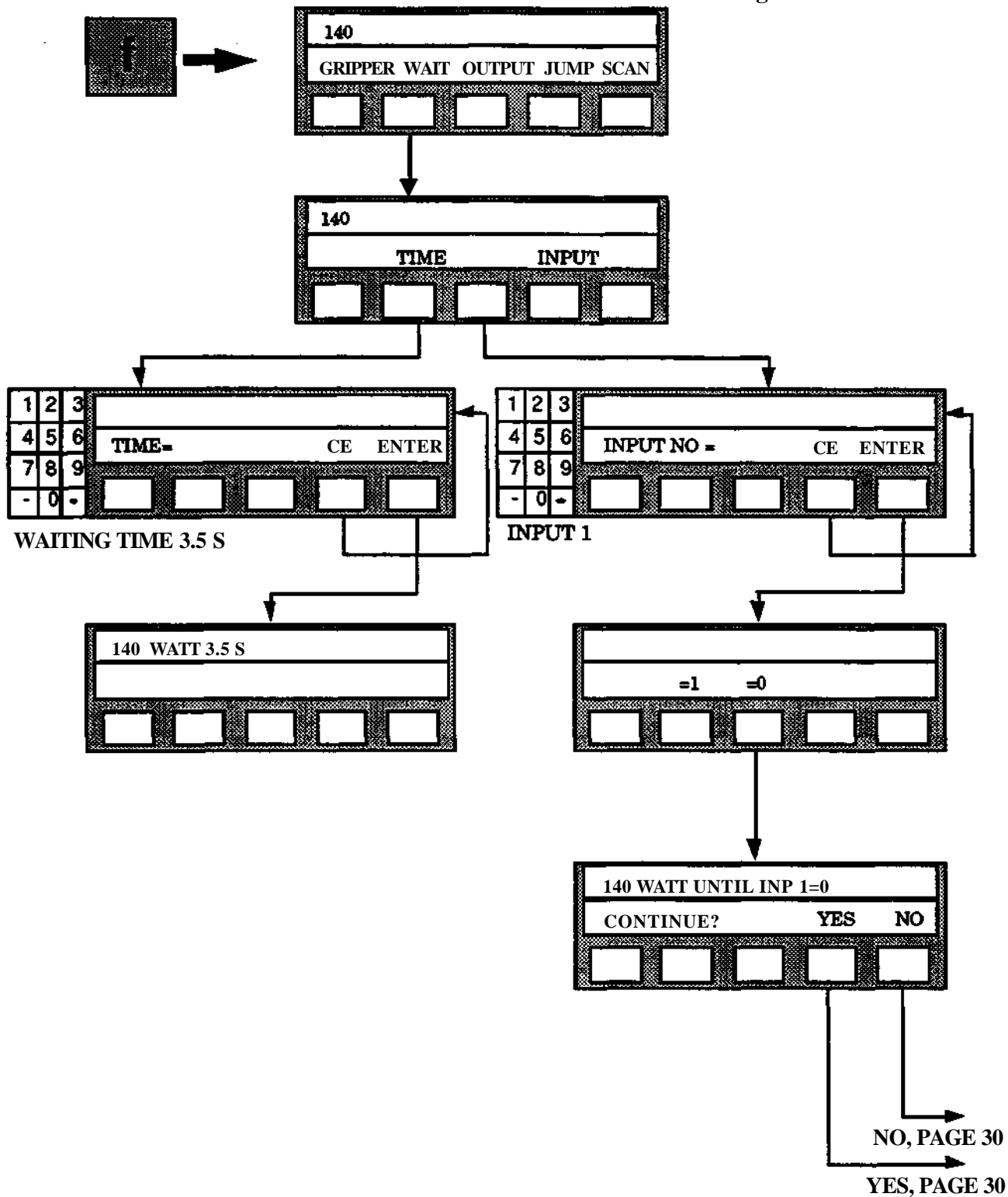
The current TRIM factor is displayed out on the topline on an optional monitor when the program stops.

The same information is displayed on the programming unit by pressing the SHIFT key.

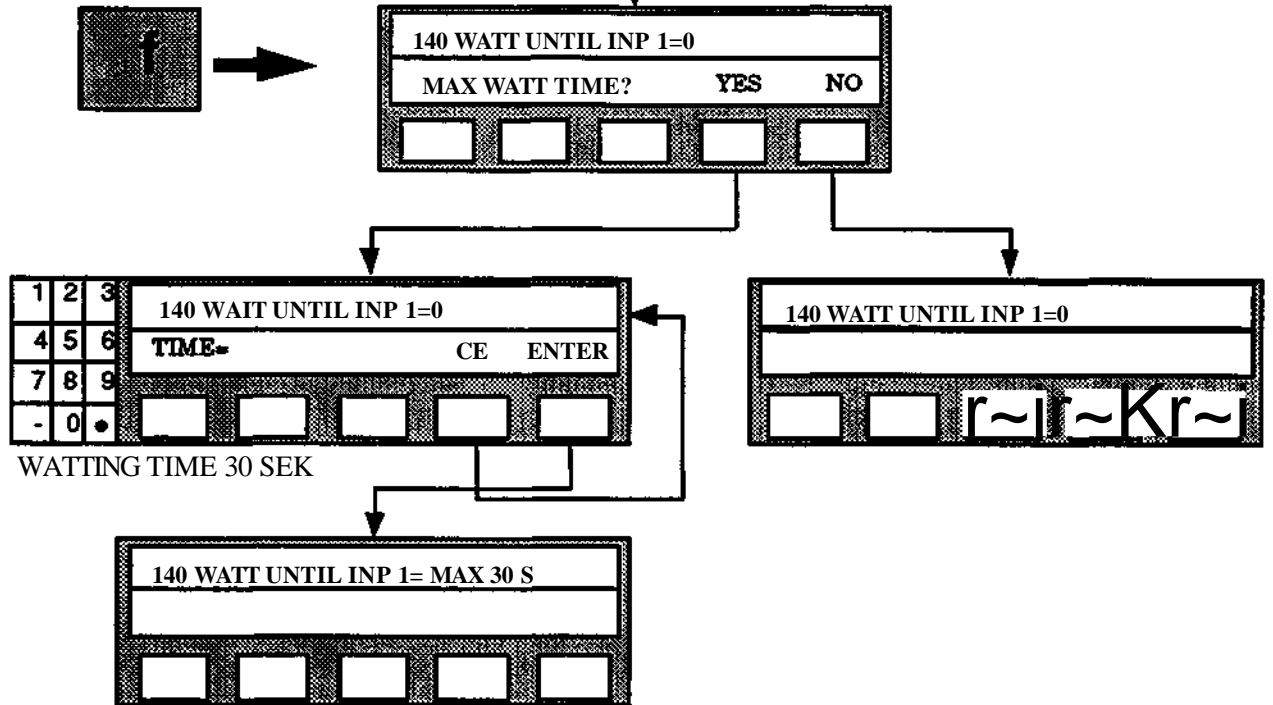
5.22**LOCATE and VISCTKL**

The instructions make it possible to program image processing instructions. For detailed information of the function and how to program, see OptiMaster User's manual.

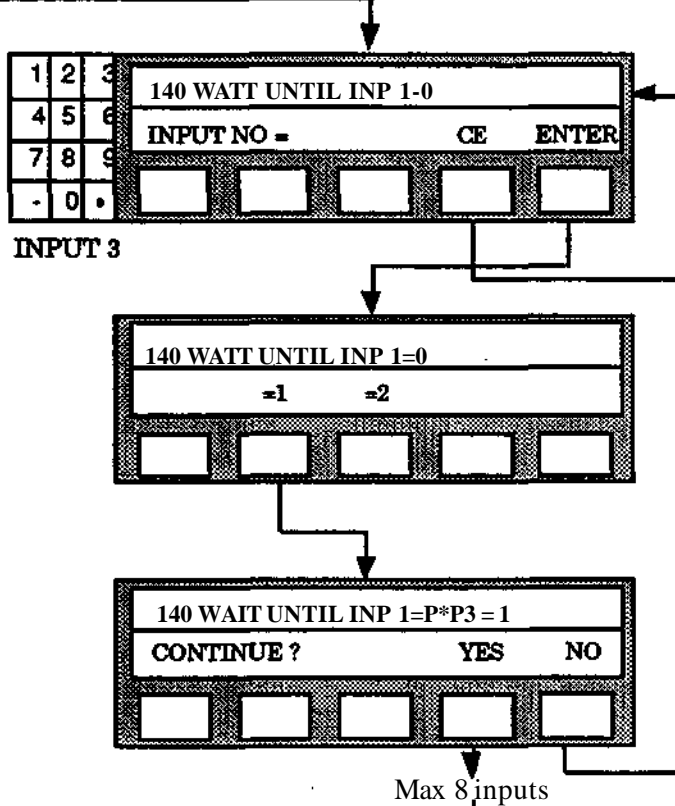


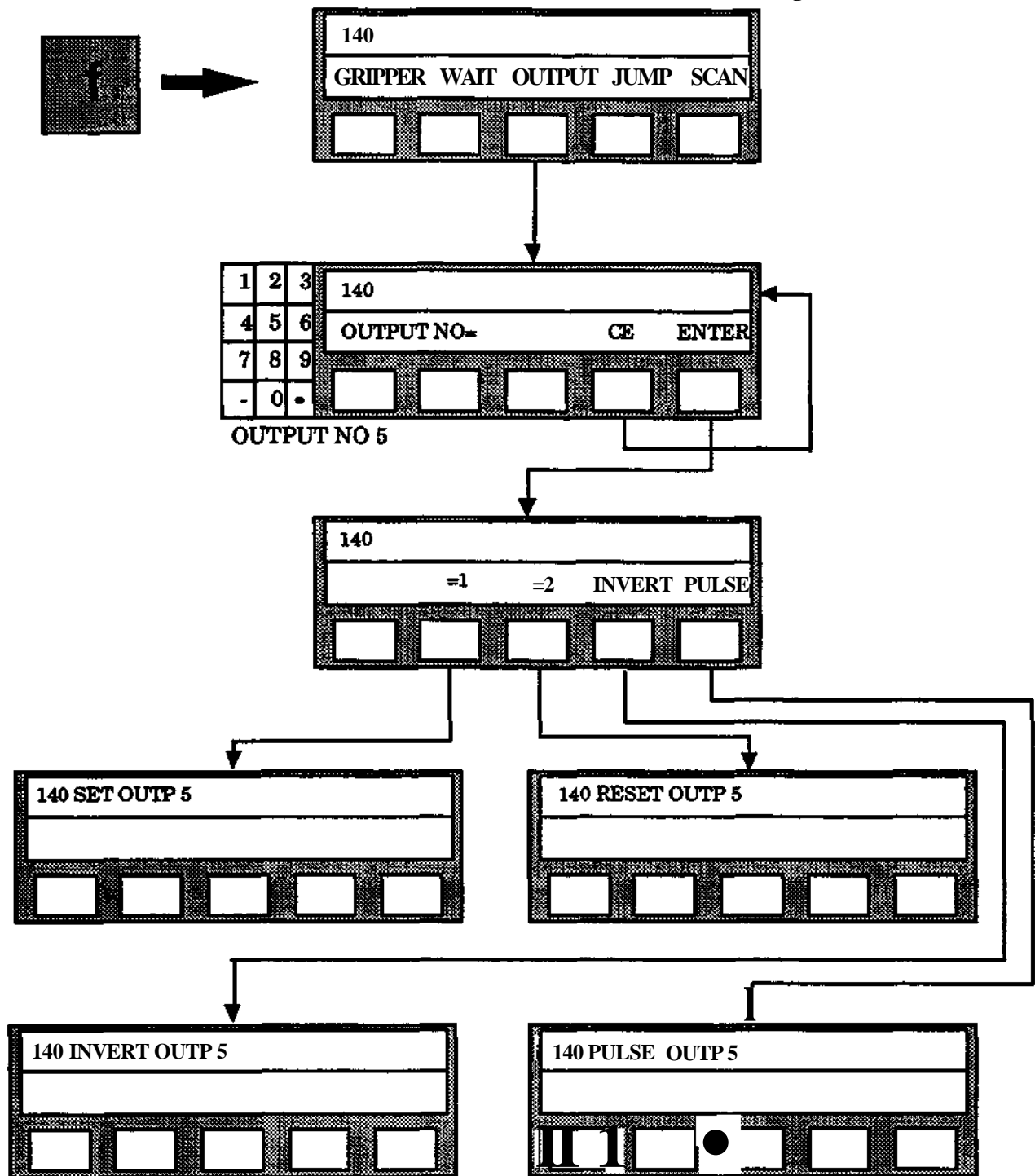


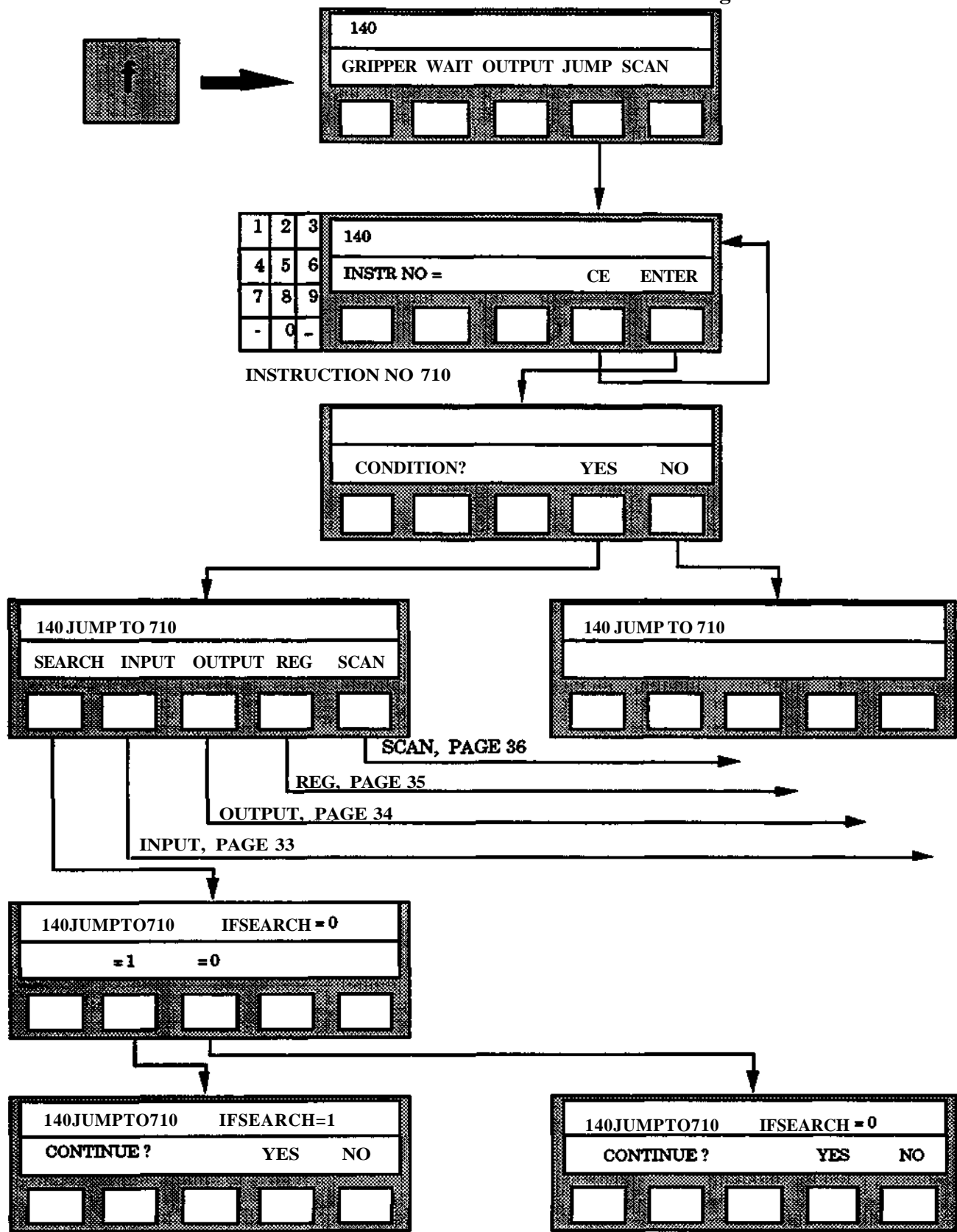
NO, PAGE 29



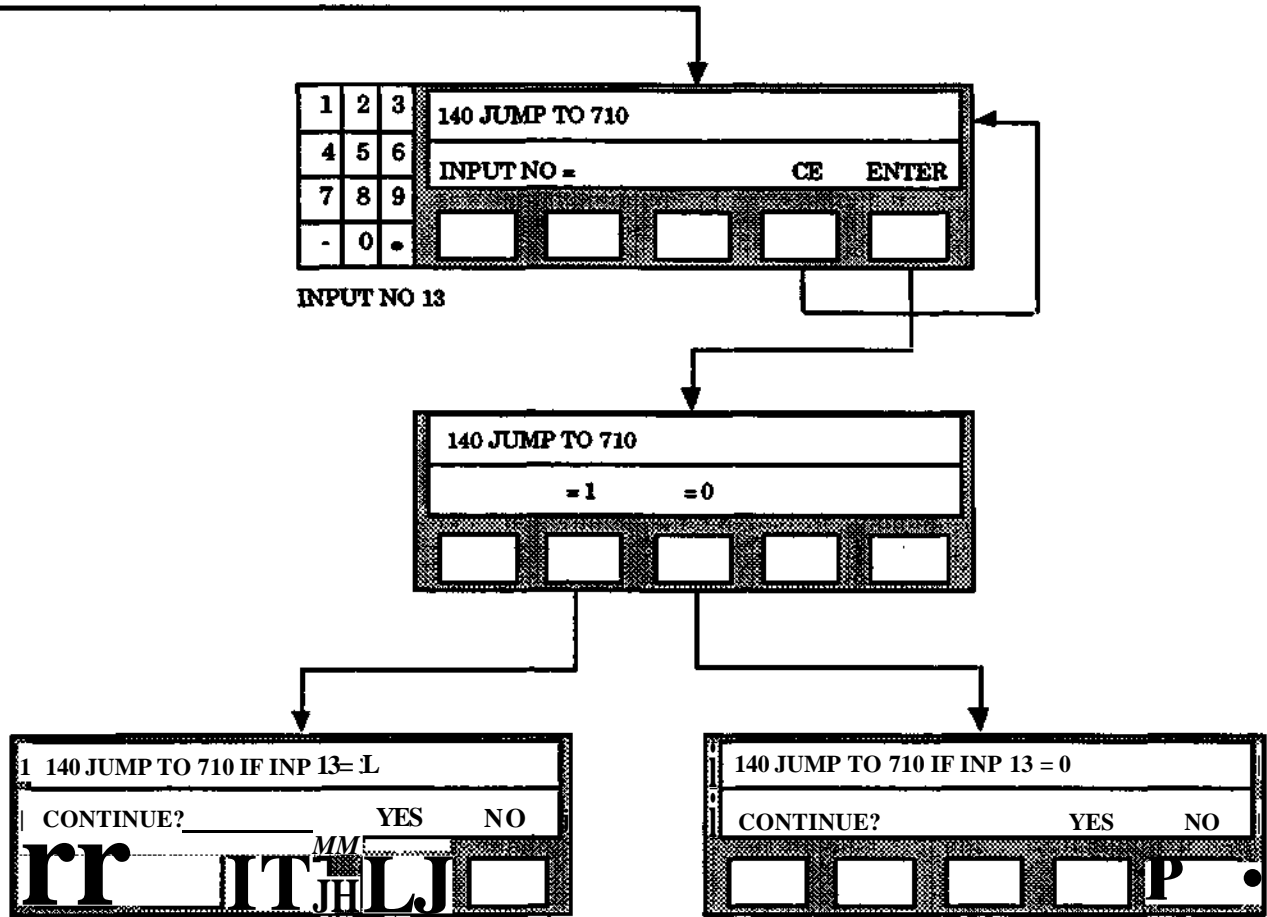
YES, PAGE 29

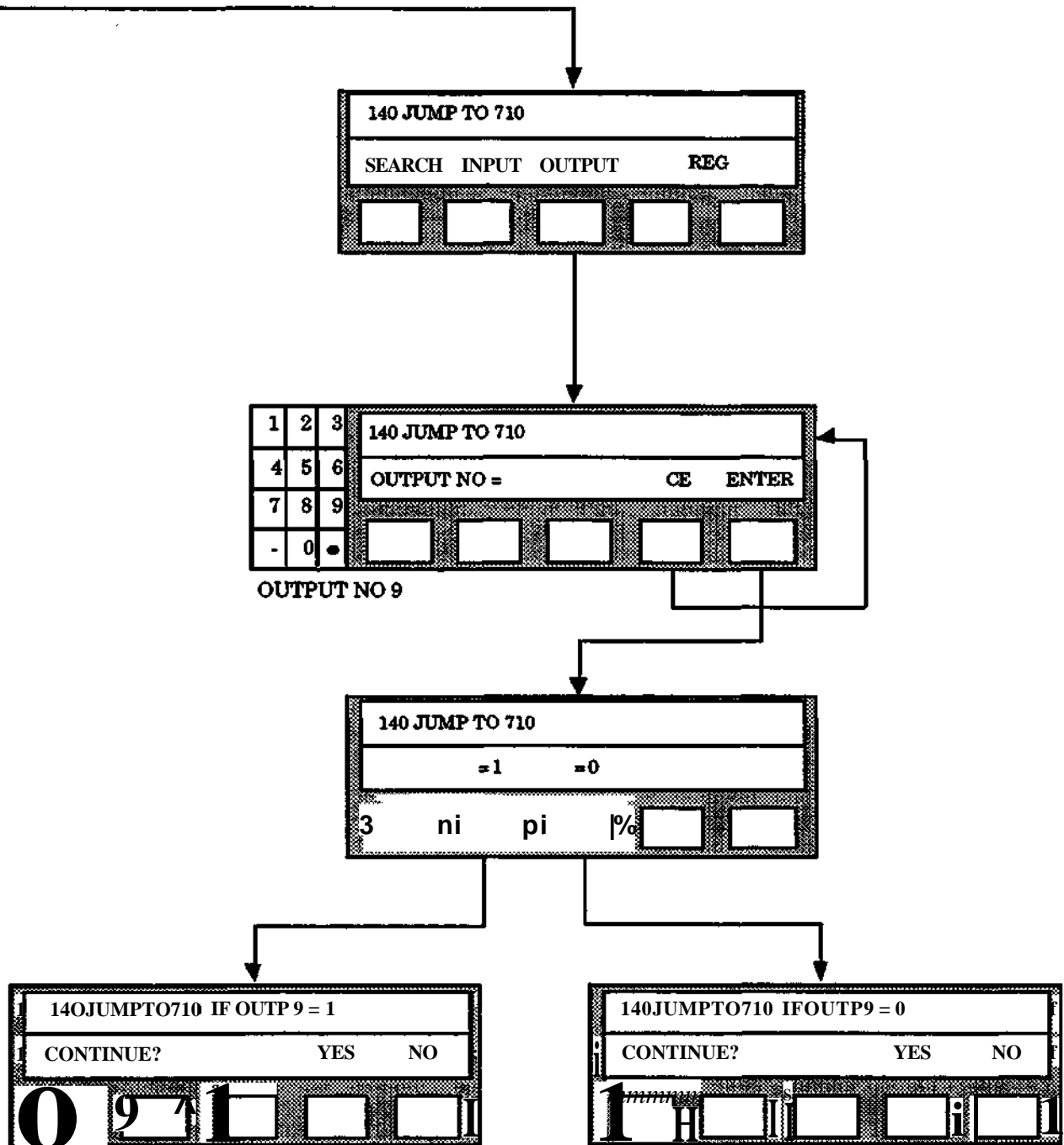




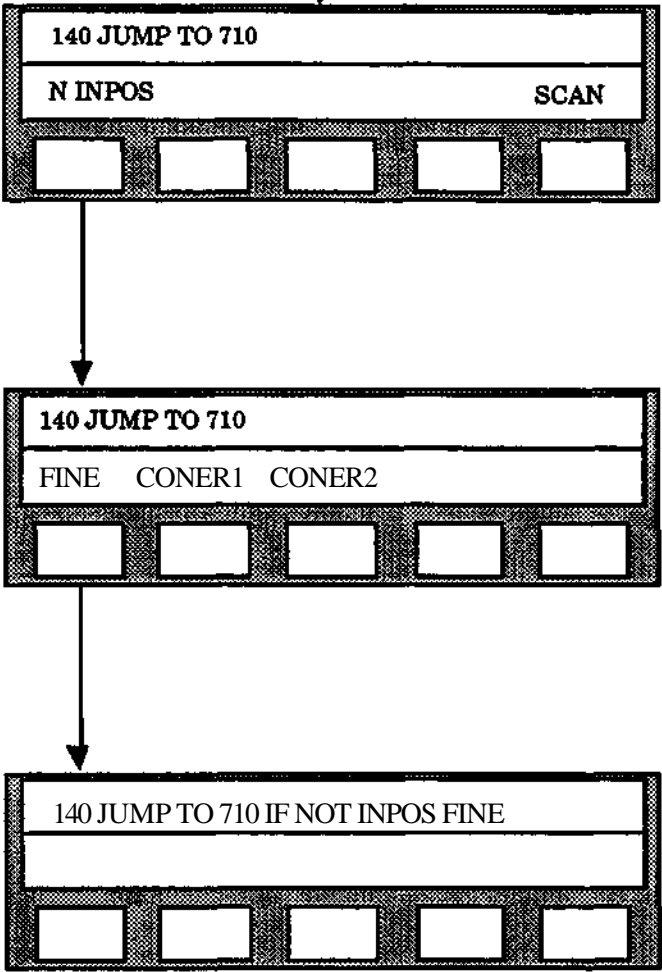


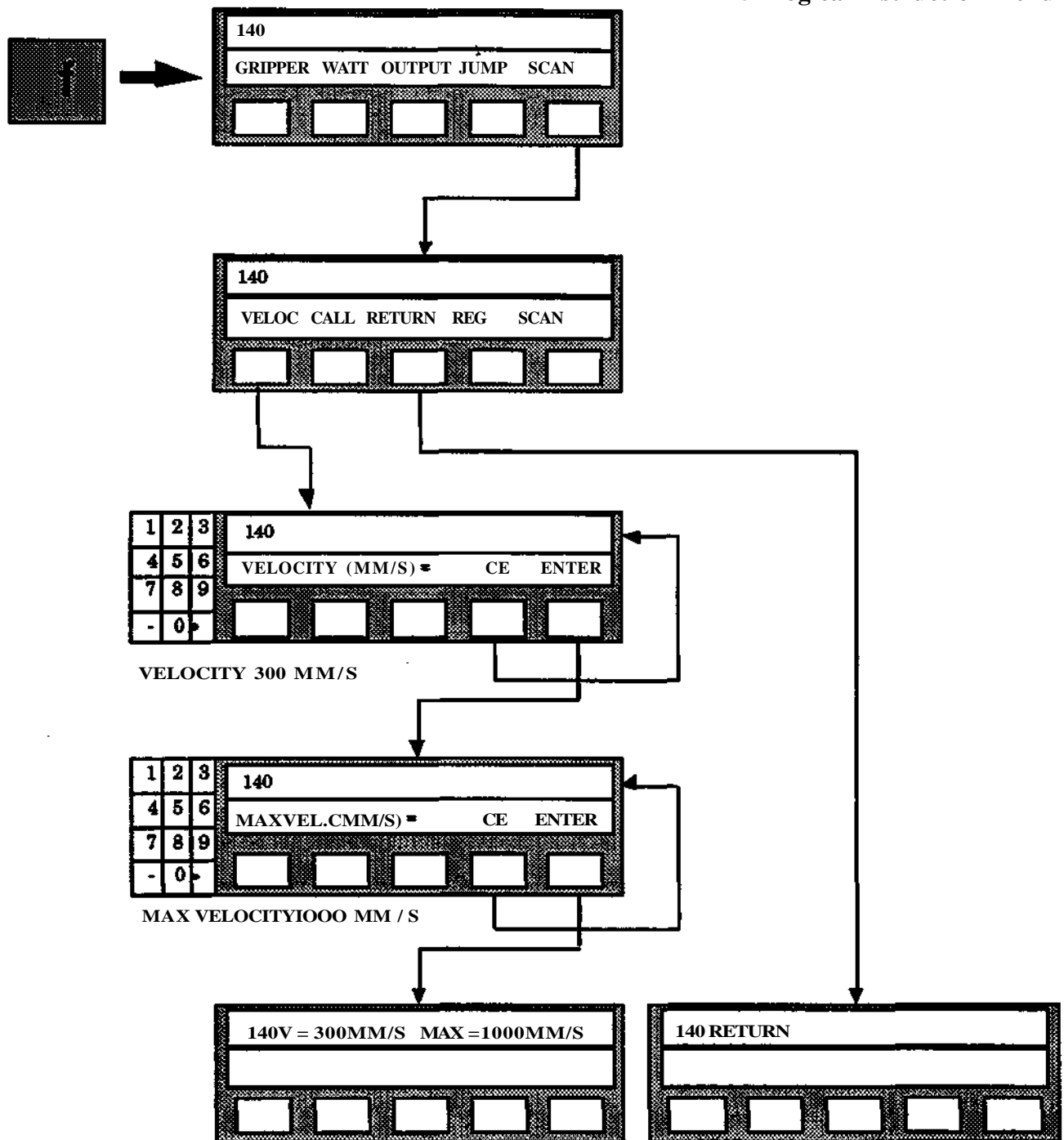
INPUT, PAGE 32

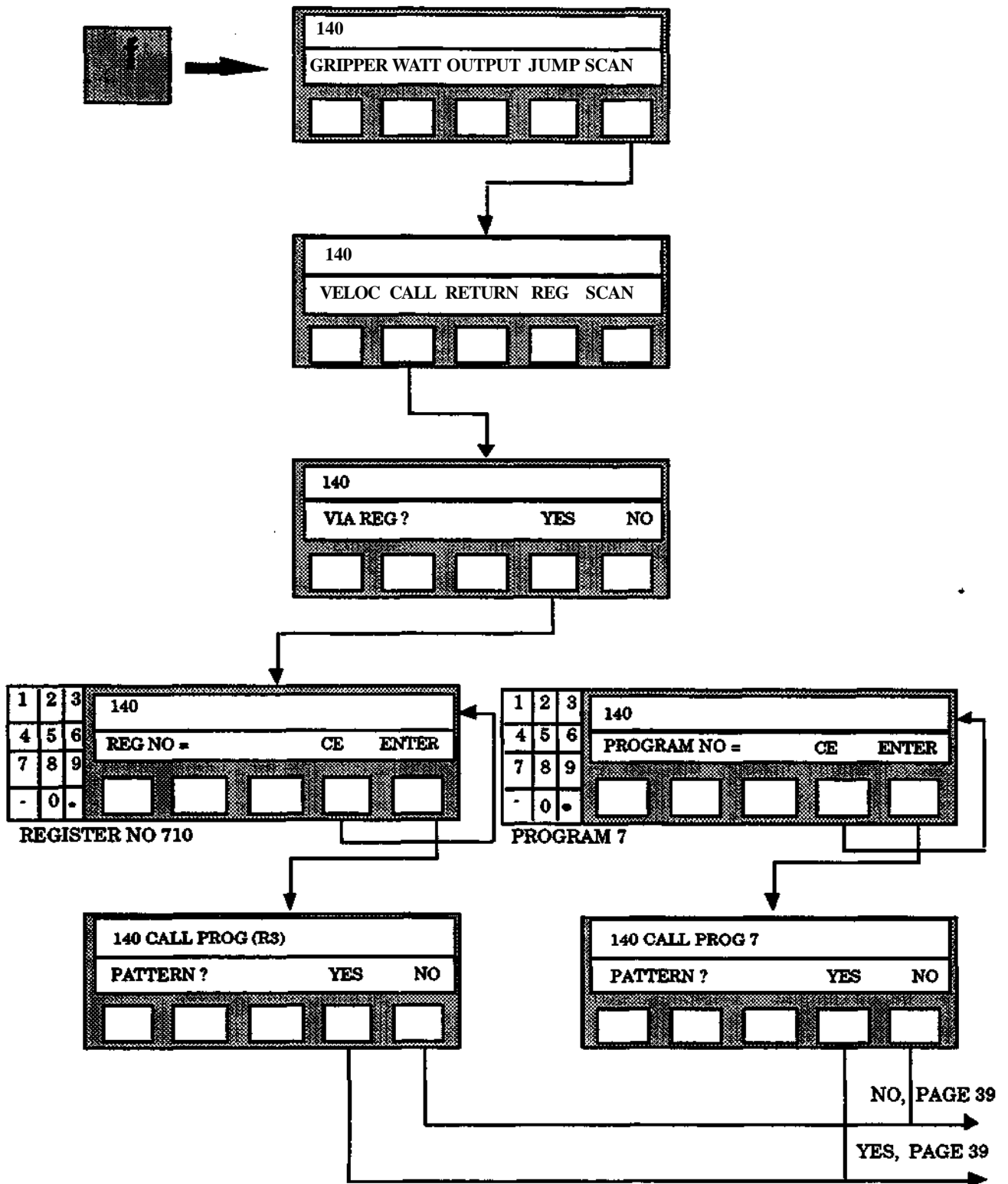




SCAN, PAGE 32







NO, PAGE 38

YES, PAGE 38

1	2	3	140		
4	5	6	REG NO =		
7	8	9	CE ENTER		
-	0	.			

REGISTER NO 5

140 CALL PROG 7..					
REPEAT ?				YES	NO

140 CALL PROG ... R5					
REPEAT ?				YES	NO

1	2	3	140 CALL PROG 7...		
4	5	6	NO OF REP =		
7	8	9	CE ENTER		
-	0	.			

3 REPETITIONS

140 CALL PROG ... R5					

140 CALL PROG 7... KEP3					

1	2	3	140 CALL PROG ... R5		
4	5	6	NO OF REP =		
7	8	9	CE ENTER		
-	0	.			

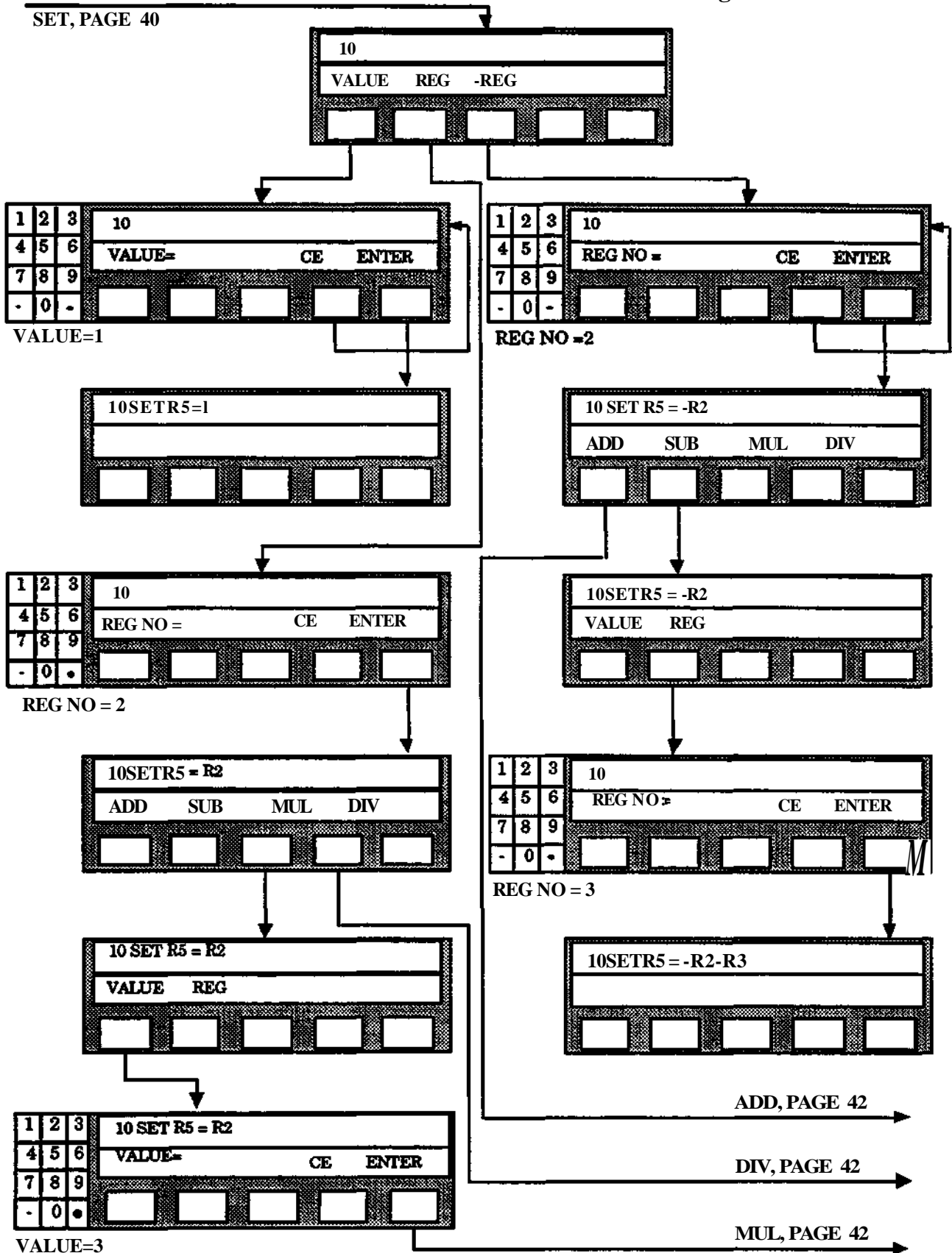
4 REPETITIONS

140 CALL PROG 7...					

140 CALL PROG... R5 REP 4					


```

graph TD
    T[T] --> S1[140  
GRIPPER WATT OUTPUT JUMP SCAN]
    S1 --> S2[140  
VELOC CALL RETURN REG SCAN]
    S2 --> S3[140  
FETCH LET TRANSF LOC]
    S3 --> S4[140  
REG NO = CE ENTER]
    S3 --> S5[140  
PORT NO = CE ENTER]
    S4 --> S6[140  
140 FETCH R5 FROM PORT NO 12]
    S5 --> S7[140  
140 TRANSFER RS TO PORT NO 3]
    S5 --> S8[140  
REG NO = CE ENTER]
    S6 --> S8
    S7 --> S8
    S8 --> S9[140  
REGISTER NO 5]
    S9 --> SET[SET, PAGE 41]
  
```

ADD, PAGE 41

10 SET R5 = - R2								
VALUE			REG					

1	2	3	10 SET R5 = - R2					
4	5	6	VALUE REG					
7	8	9						
-	0	-						

REG NO 3

10 SET R5 = - R2 + 3								

DIV, PAGE 41

10 SET R5 = - R2								
VALUE			REG					

1	2	3	10 SET R5 = - R2					
4	5	6	VALUE REG					
7	8	9						
-	0	-						

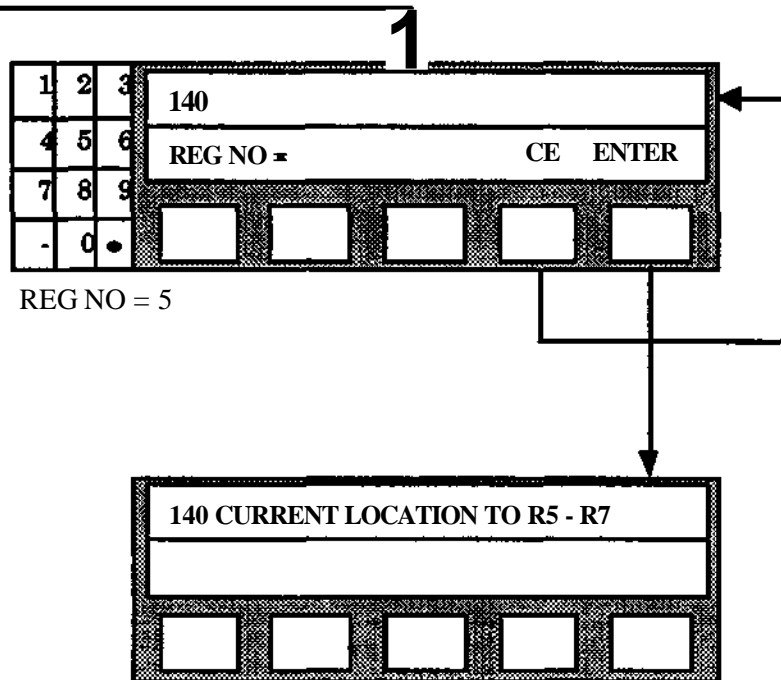
REG NO 3

10 SET R5 = - R2 / R3								

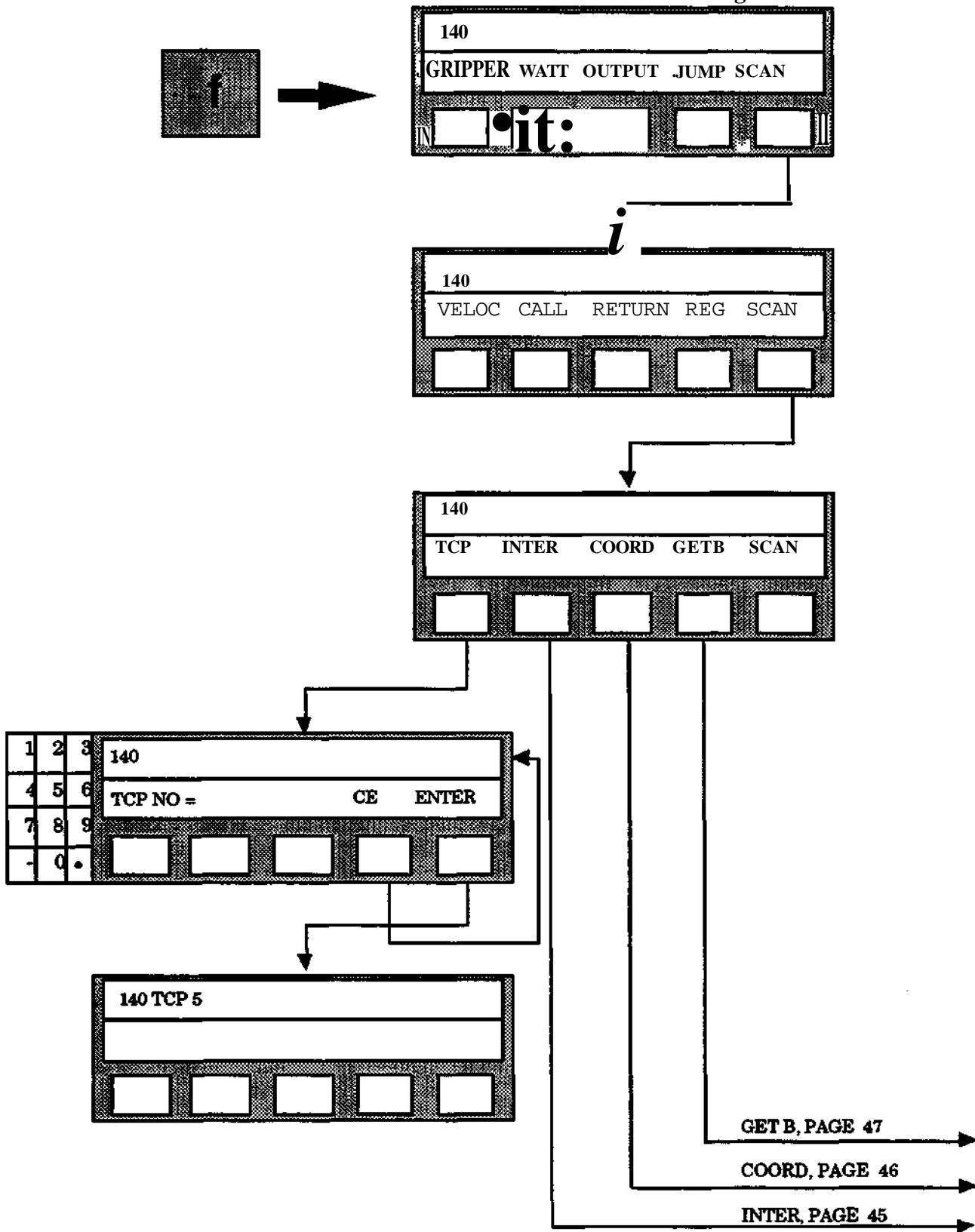
MUL, PAGE 41

10 SET R5 = R2 * 3								
VALUE			REG					

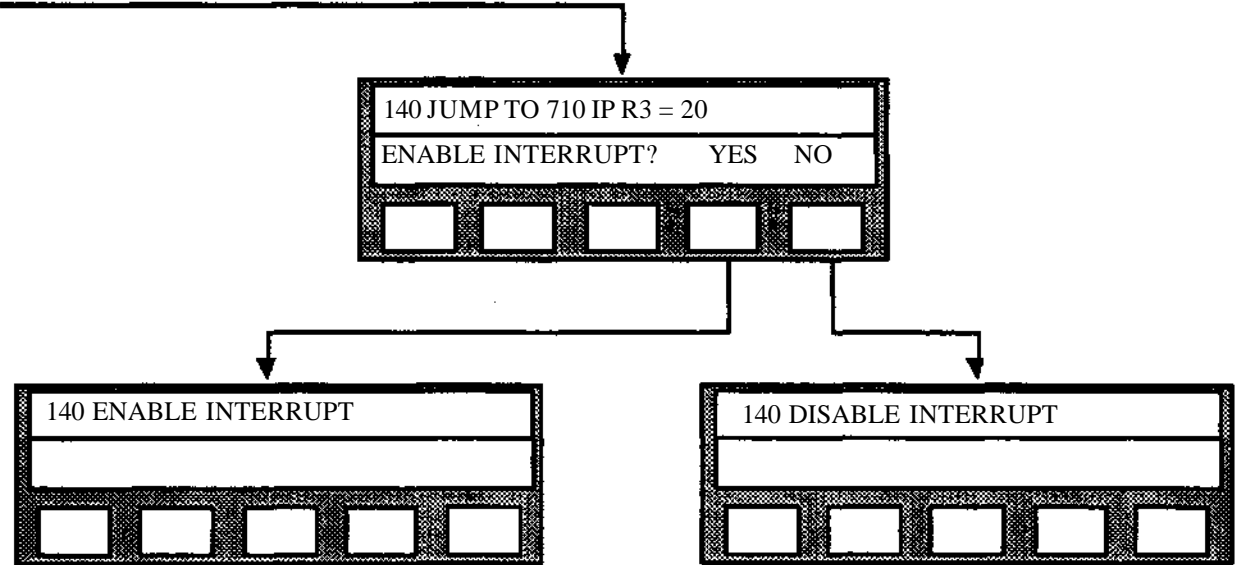
POS, PAGE 40



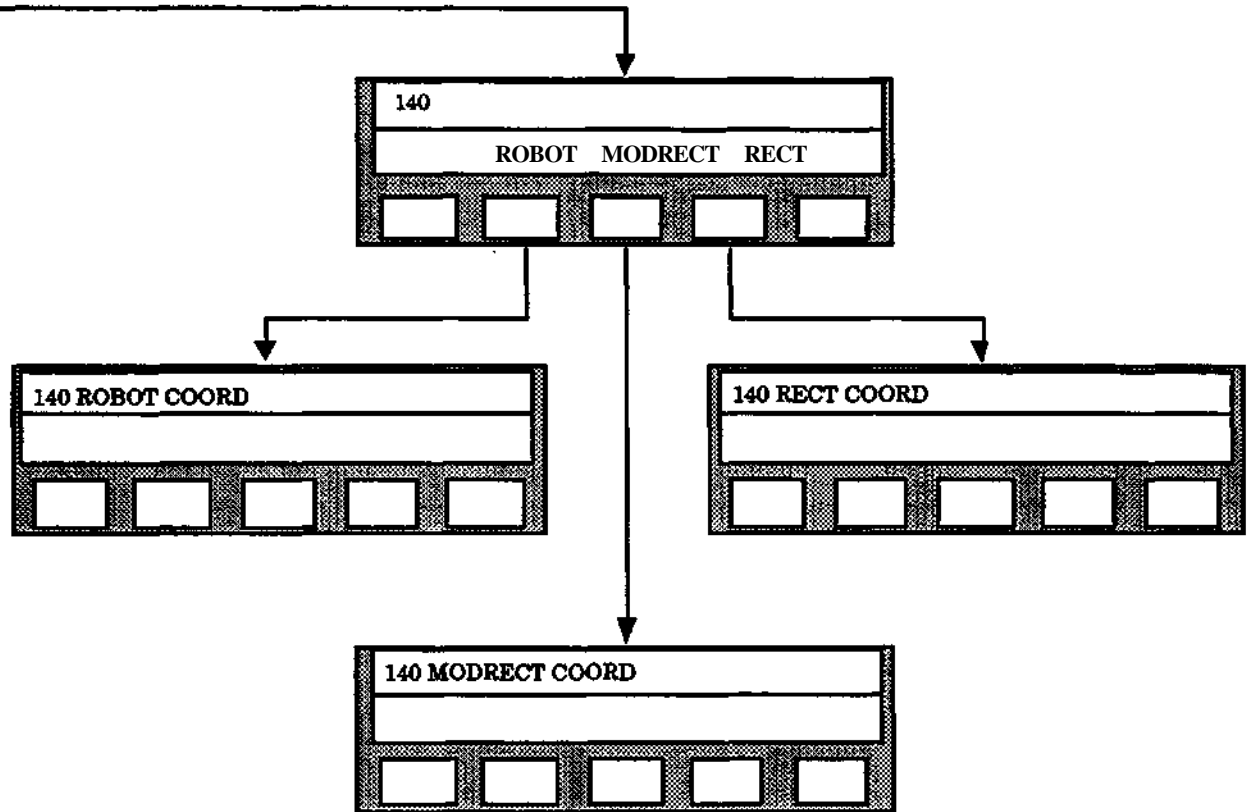
5 Logical instruction menu

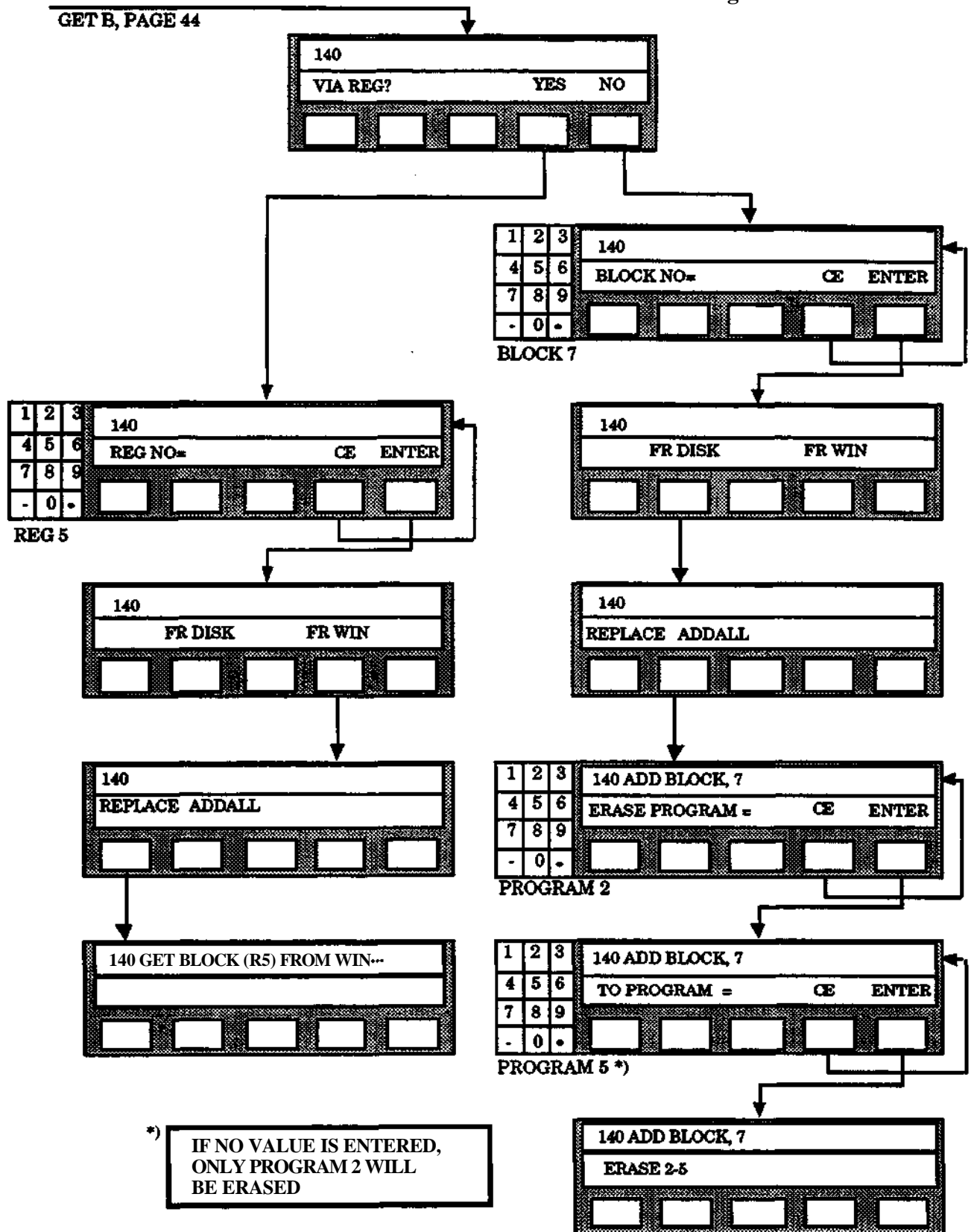


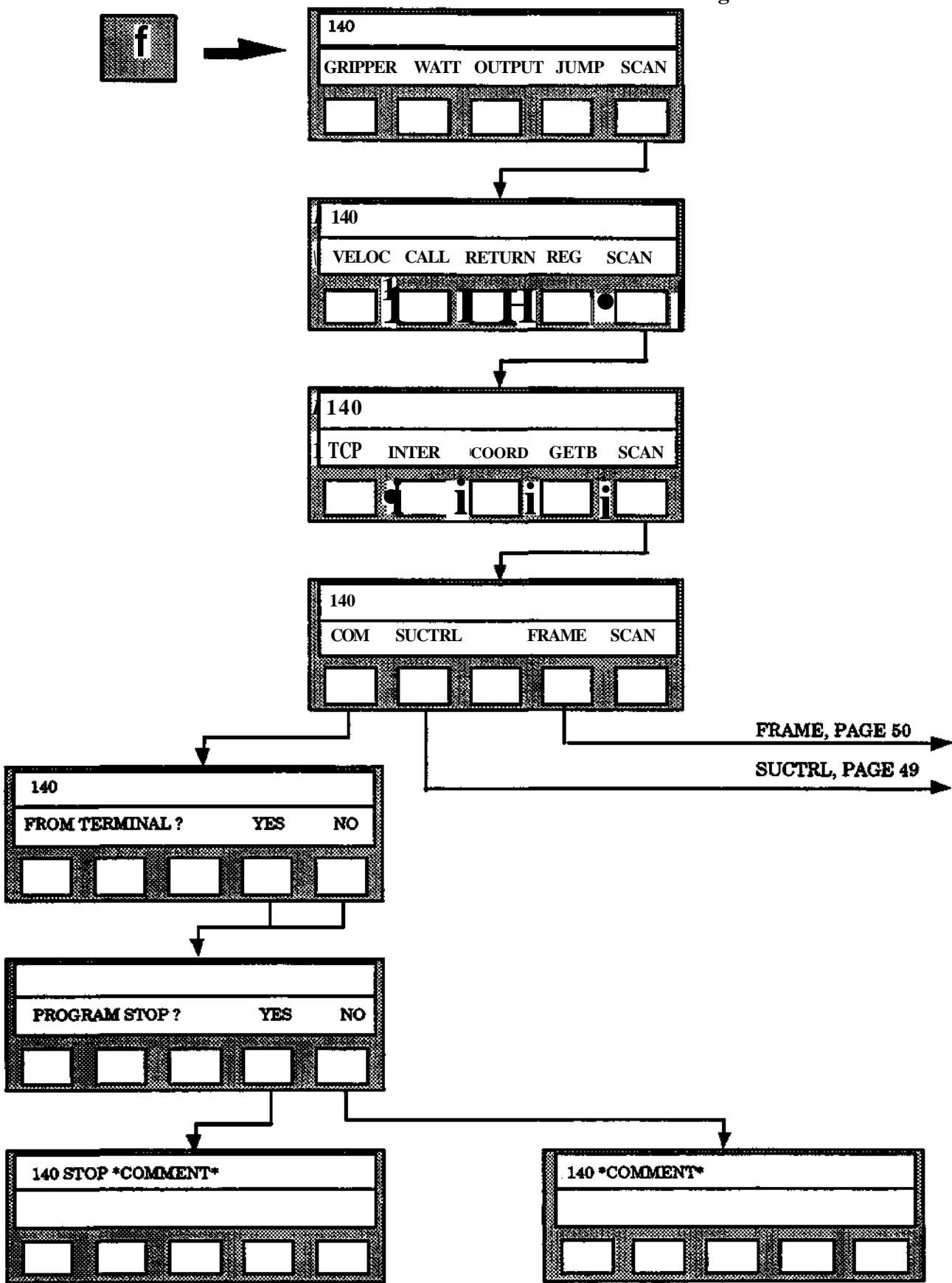
INTER, PAGE 44

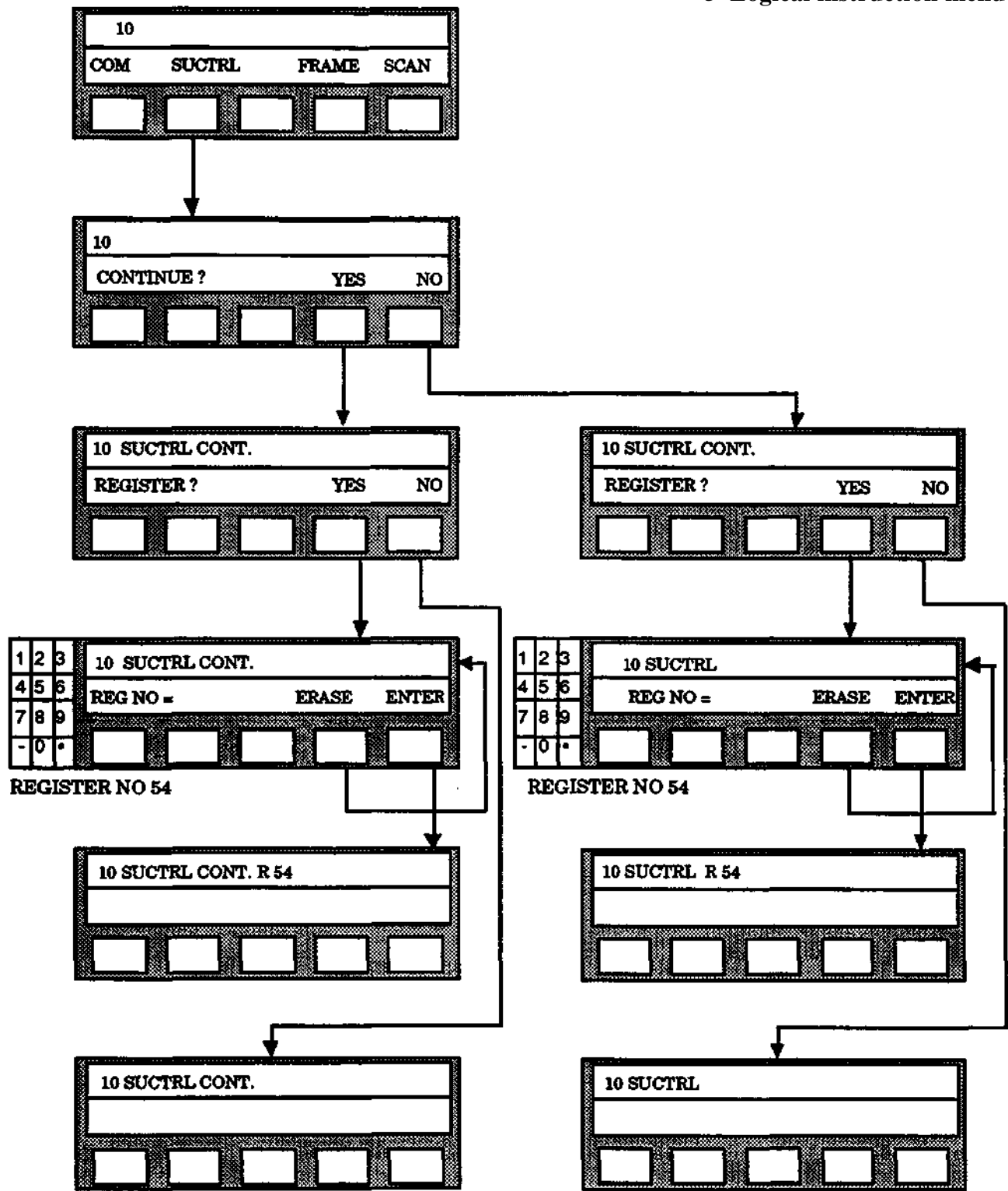


COORD, PAGE 44

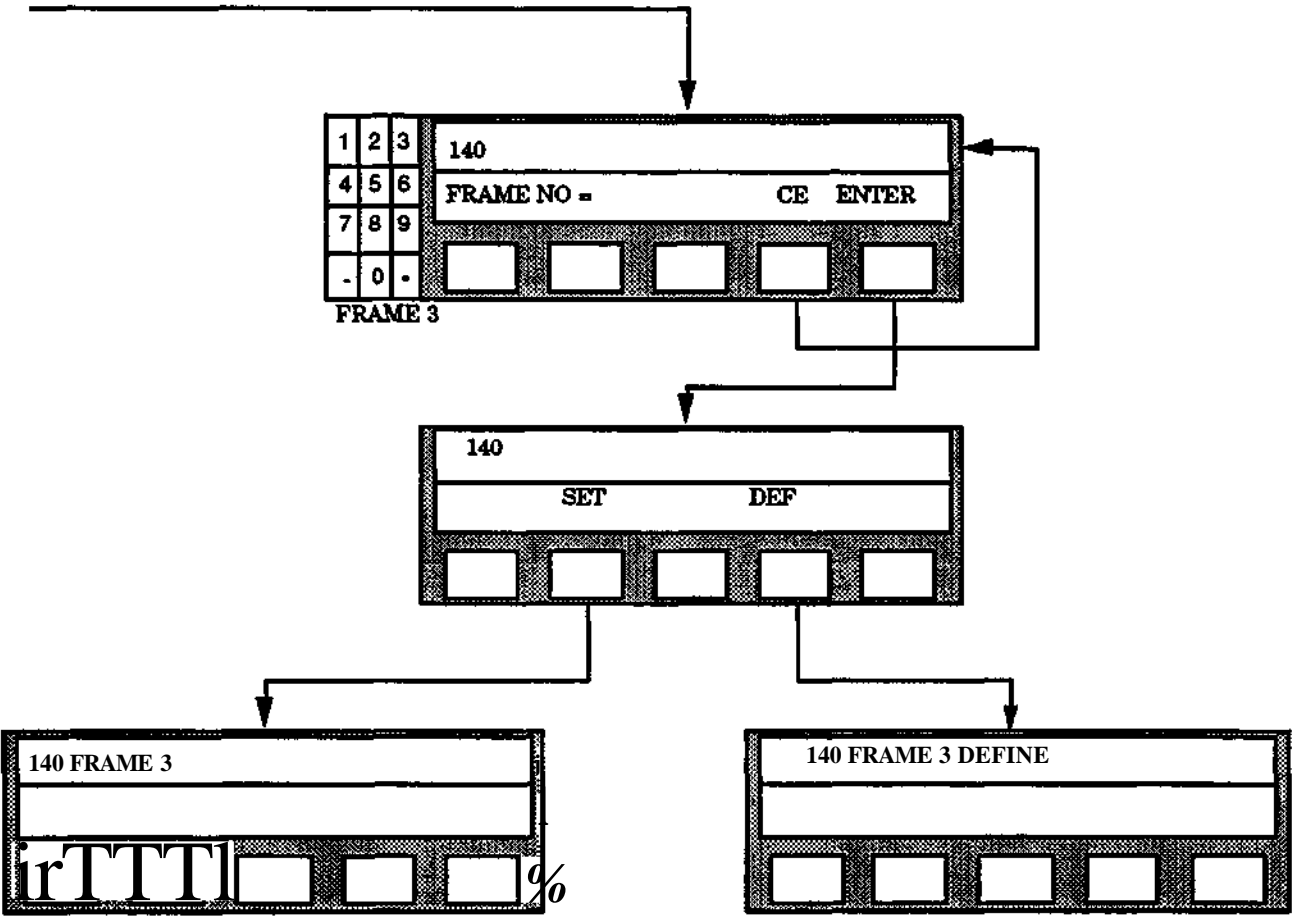




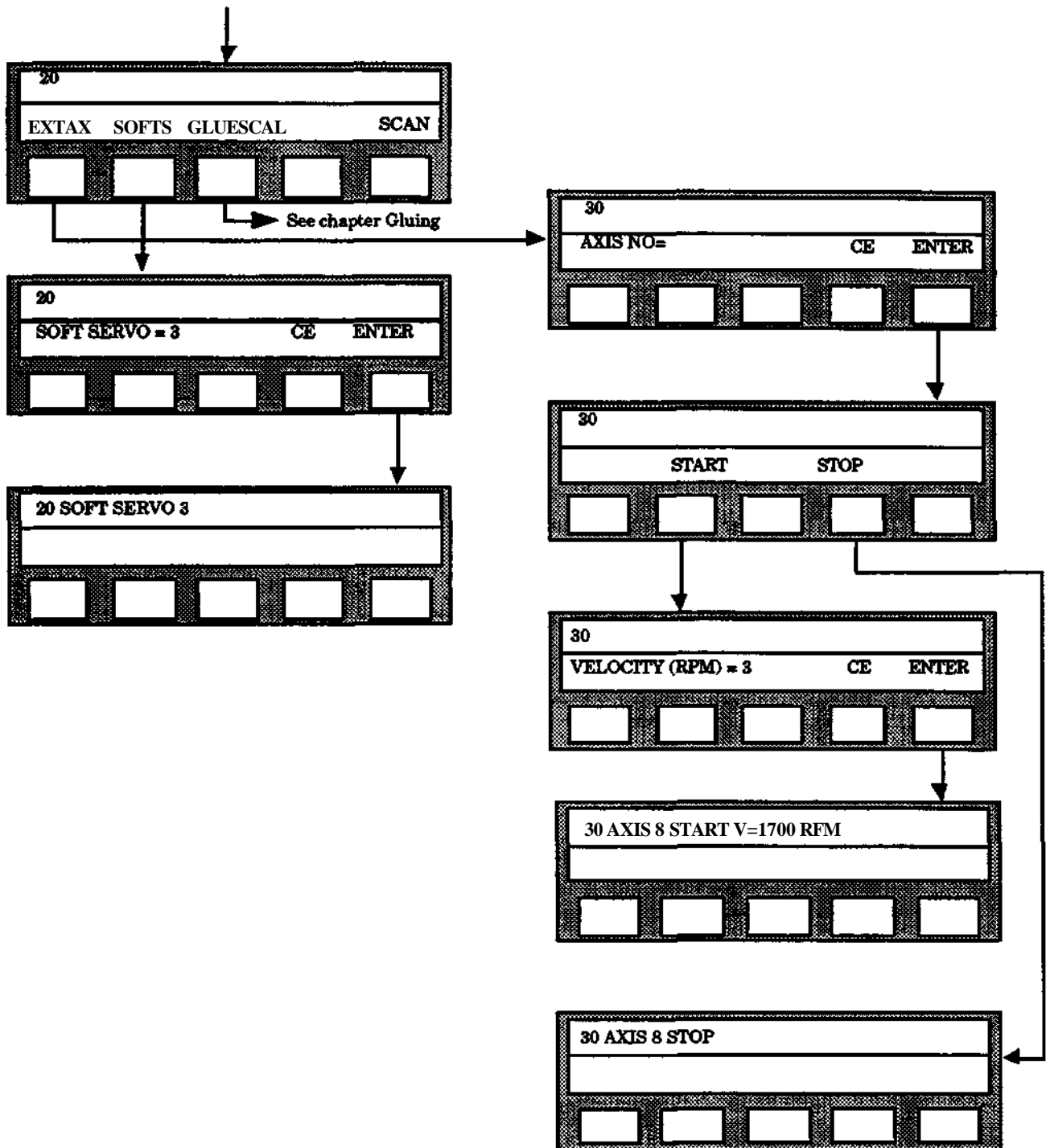


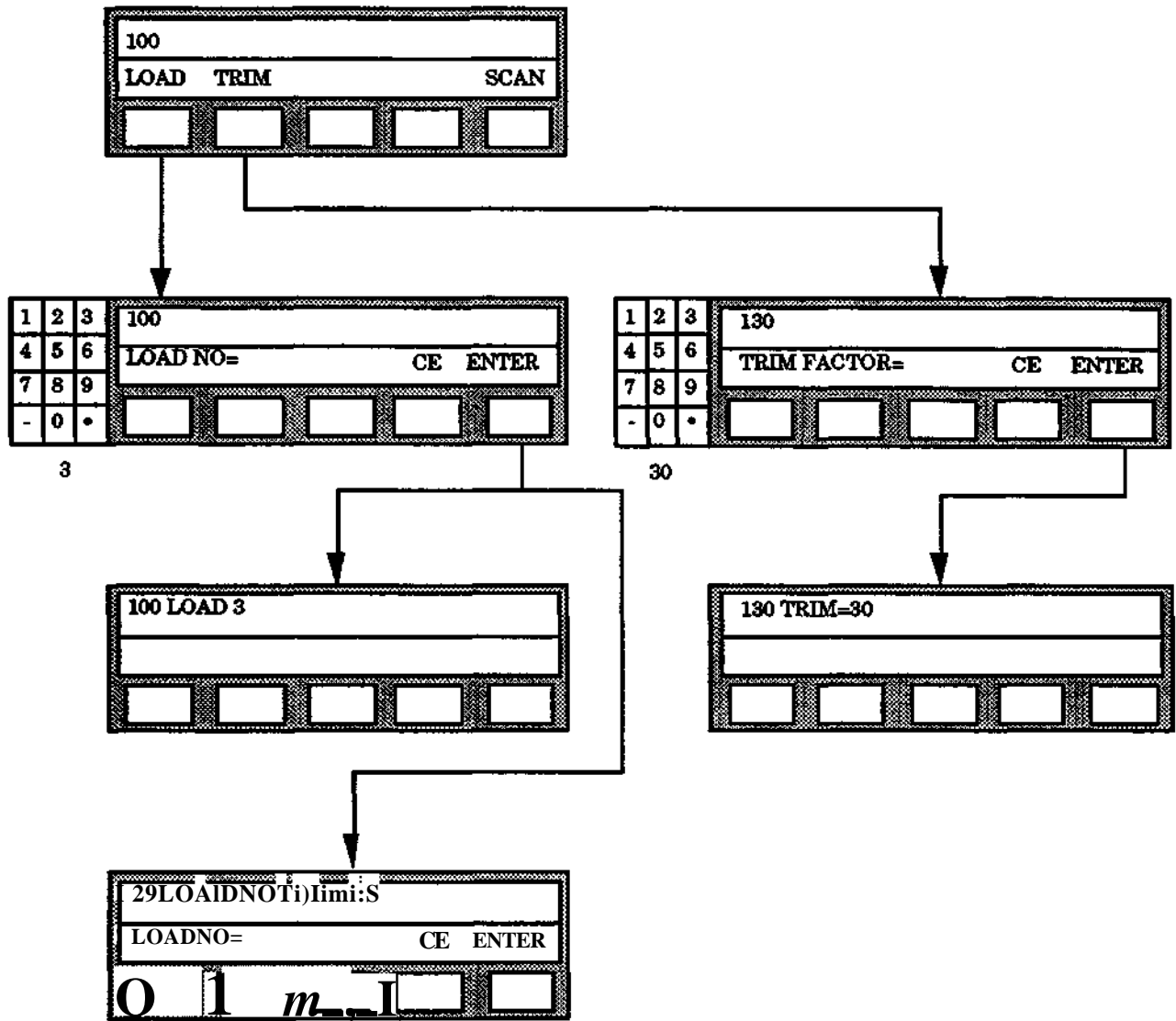


FRAME, PAGE 48

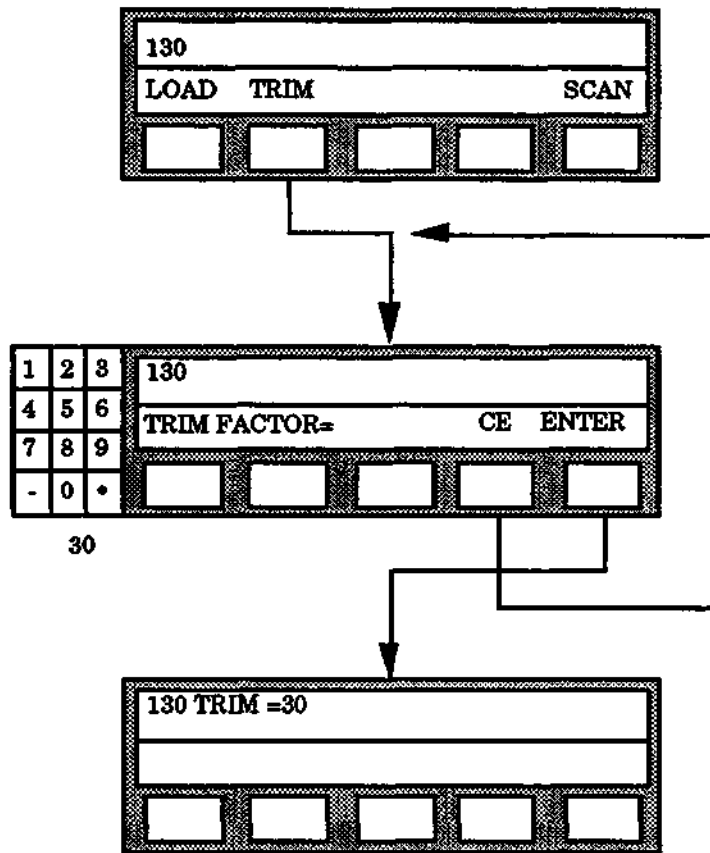


 → SCAN 4 TIMES





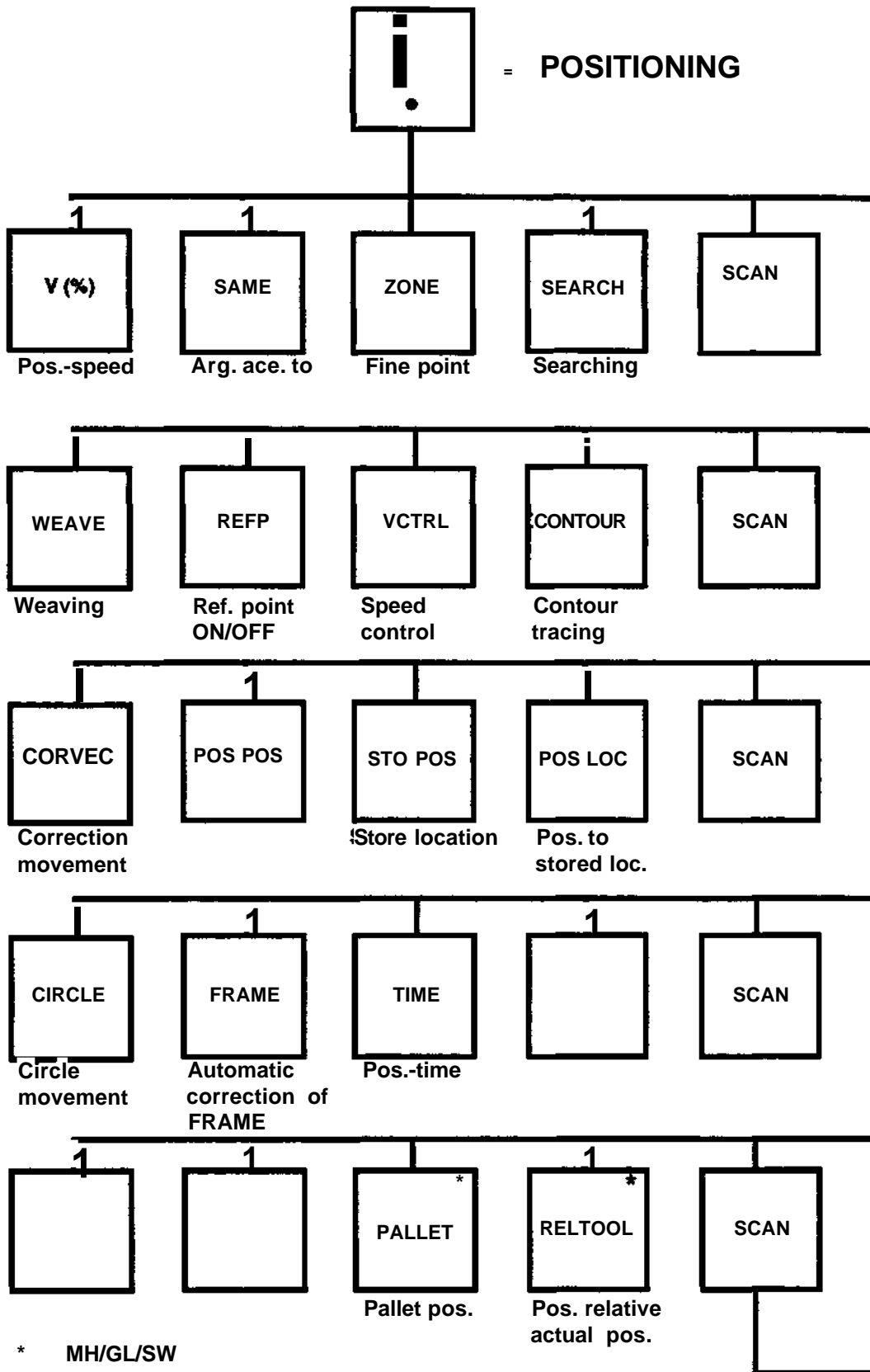
FUNCTION + SCAN 5 TIMES





6. Position instruction menu

Section	Page
6.1 V%	6:5
6.2 SAME	6:5
6.3 ZONE	6:5
6.4 SEARCH	6:6
6.4.1 SEARCH (dist)	
6.4.2 SEARCH (dir)	
6.4.3 SEARCH (auto)	
6.5 WEAVING	6:8
6.6 REFP	6:12
6.7 VCTRL	6:12
6.8 CONTOUR	6:13
6.9 CORVEC	6:14
6.10 POSPOS	6:14
6.11 STOPOS	6:16
6.12 POSLOC	6:17
6.13 CIRCLE	6:18
6.14 FRAME	6:18
6.15 TIME	6:19
6.16 PALLET	6:19
6.17 RELTOOL	6:21
6.17.1RELTOOL (X, Y, Z)	
6.17.2RELTOOL (via REG)	
6.18 EXTFRAME	6:23



6 Position instruction menu



6. POSITIONING

Programming of position

The Menu button gives the instruction

1. Position the tool to the required position with the joy stick
2. Press the POSITION menu button. The position is programmed and the percentage speed which applies is shown.

If the percentage speed is not to be changed omit the next procedure. If the position is only to be a PATH point, the programming of the position is now complete.

Note

When 10 -12 axes are present the system generates two instructions, according to the following example:

240 AXES 10 -12 POS

250 POS V = 100% PATH

Extra instruction

Normal instruction

6.1

$$V_0$$

Change of percentage velocity

1. Select V % in the menu
2. Specify the percentage speed 0-799.9 %.

6.2

SAME

If the complete instruction is to be repeated in a new position, select SAME in the menu and conclude in this way the programming of the position.

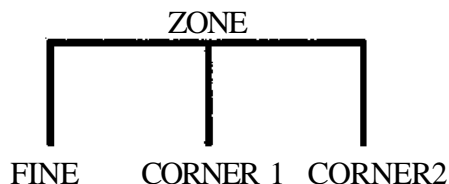
In other cases, the definition of the zero zone is described in the next procedure whereas other arguments described briefly under "Otherwise".

6.3

ZONES

There are four alternatives for programming a zone. The values for the alternatives are defined as system parameters. The default zone alternative for an instruction is PATH. The other alternatives are programmed by using the ZONE button:

Three alternatives will appear:



The text on the display shows which alternative is active:

100 POS V = 100 % PATH

100 POS V = 100 % FINE

100 POS V = 100 % CI

100 POS V = 100 % C2

The default values of the system parameters have been chosen to provide a useful choice of zone sizes. Particular applications may require other values to be defined, see chapter 11.

ZONE	Default value
PATH	-2 Note! ("-") indicates velocity dependent zone size
CORNER1	15
CORNER2	-100
FINE	2

Use of zones	
PATH	for normal path
CORNER1	for corners with small deviation, regardless of speed
CORNER2	for big corner rounding and thus high speed
FINE	for precise positioning and when logical instruction must be carried out at the position.

See section 3.6.1 for a more detailed description of corner handling.

6.4 SEARCH

See section 10.3 for more detailed description.

6.4.1 SEARCH (DIST)

Program a distance searching as follows:

1. Program a FINE position instruction
2. Press SEARCH
3. Press DISTANCE
4. Specify the number of the sensor (1-16) with the numerical button set.
5. Press ENTER. If the question "STOP CONDITIONS=" is shown on the lower line of the display, continue with point 6. Otherwise continue directly to point 8.
6. Specify the search stop condition as a percentage (0-100) of the maximum signal range of the sensor with the numerical button set. If the current signal level is to be a stop value, run the robot with the joy stick to the position where the sensor gives the required signal.
7. Press ENTER.
8. If several sensors are to be used press YES and repeat the points 4-6 for the new sensor. Otherwise press NO and continue with point 9.
9. If a delayed search stop is required, press the function button YES. Otherwise press NO.

The argument SEARCH is now programmed. If required, more arguments can be added to the instruction.

6.4.2 SEARCH (DIR)

Program a direction searching as follows:

- 1 Program FINE position instruction.
- 2 Press SEARCH
- 3 Press DIR
- 4 Enter the number of the sensor (1-16) with the numerical button set
- 5 Press ENTER. If the question "STOP CONDITIONS=" is shown on the lower line of the display, continue with point 6. Otherwise proceed to point 8.
- 6 Specify the search stop condition, as a percentage (0-100) of the maximum signal range with the numerical button set. If the current signal level is to be a stop value, run the robot with the joy stick to the position where the sensor gives the required signal.
- 7 Press ENTER.
- 8 If more sensors are to be used, press YES and repeat the points 4-6 for the new sensor. Otherwise press NO and continue with point 9.
- 9 If a detailed search stop is required, press function button YES. Otherwise press NO.

The SEARCH argument is now programmed completely. If required, more arguments can be added to the instruction.

6.4.3 SEARCH (AUTO) AW only

Program automatic distance searching AUTO SEARCH as follows:

Program the start position in accordance with "Distance search". Then jog the robot to the position close to the intended search stop position:

- 1 Program a FINE position instruction.
- 2 Press SEARCH.
- 3 Press AUTO.
- 4 Specify the number of the sensor (1-16) with the numerical keyboard.
- 5 Press ENTER.
- 6 Answer the question "MORE SENSORS?". If YES, return to point 4. Otherwise, continue as below.
- 7 Answer the question "DELAY?". The instruction is now complete.

The resulting instruction looks as the example below:

140 POS V = 100% AUTOSEARCH SI
where "SI" is the sensor used.

6.5 Weaving MH/GL/SW not IRB 6000

In AW robot the function is replaced by the integrated weaving function which is part of the welding instruction.

Menu: POSITIONING

Function: WEAVE

Means:

A periodic recalling transverse movement (weaving), stored in a separate sub-program, the weaving program is superimposed on a movement to a programmed position, the main movement.

Facts:

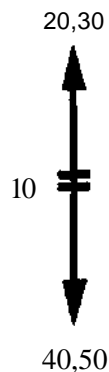
The weaving movement is a combination of:

- The main movement between the start point and the end point, in the ordinary program.
- A superimposed periodic weaving movement, in a separate weaving subprogram.

The example below shows how each of the programs contributes with one part of the resulting weaving movement.

Example

The movement below is performed during running of subprogram 10, with the help of weaving subprogram 200, where the periodic weaving movement is programmed.



Subprogram 10

```

.
.
90 POS V = 10% WEAVING PROG 200 H = 100%
.
.

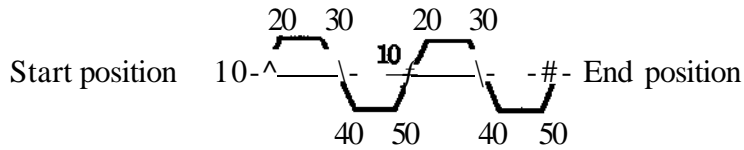
```

Subprogram 200 (weaving program)

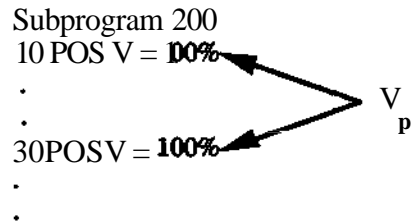
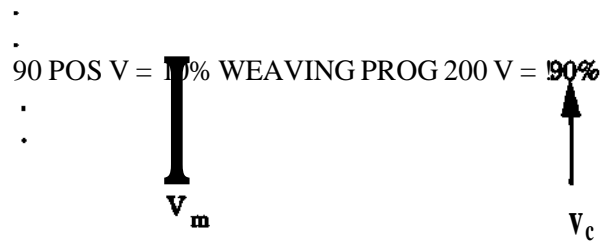
10 POS V = 100%	(Movement sideways to the left end position)
20 WAIT 1S	(Paus in the left end position)
30 POS V = 100%	(Movement sideways to the right end position)
40 WAIT 1 S	(Paus in the right end position)
50 RETURN	(Program end)

6 Position instruction menu

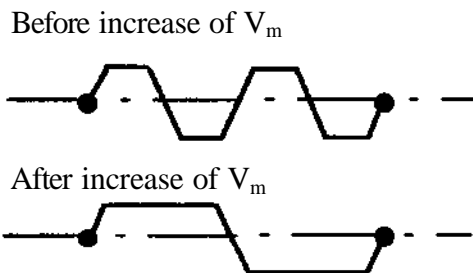
The resulting weaving movement will look as below:



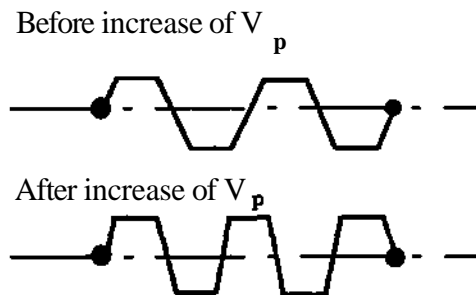
Subprogram 10



V_m = The percentage speed for the main movement in subprogram 10, instruction 90. (An increase of the percent value results in a smaller number of weavings on a specified distance. The weaving pattern is "stretched out" in the direction of the main movement and keeps it's proportions.)

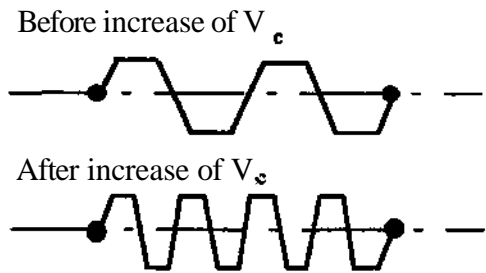


V_p = The percentage speed in the weaving program 200 for all movements. (An increase of the percent value results in faster movements sideways. On the other hand, the tool will remain in the end positions during the same time as before. The weaving pattern is "shortened" in the direction of the main movement, but will not keep it's proportions.)



6 Position instruction menu

Vc = The correction factor, expressed in percent, for how fast the weaving pattern is to be performed relative to it's own programmed speed. (An increase of the percent value gives a greater number of weavings on a specified distance. The weaving pattern is "shortened" in the direction of the main movement and will keep it's proportions.)



Weaving is performed in basic coordinates, i.e. the tool does not change orientation. It means also that the same movement pattern, performed in a new direction, requires a special weaving program. See the example below:

	1	2	3	4
Movement				
Main movement				
Result				

Weaving program

10POSV = 100%	10POSV = 100%	10POSV = 100%	10POSV = 100%
20POSV = 100%	20POSV = 100%	20POSV = 100%	20 WATT 1 S
30 RETURN	30 RETURN 100%	30POSV = 100%	30POSV =
		40 RETURN	40 WATT 1 S
			50 RETURN

Executed:

The main movement and the weaving program start at the same time to the programmed position. The resulting motion is concluded:
Within the zone for the main movement

**Note.**

Weaving can not be combined with movements of an external axis.
External axis can not be used in the weaving pattern.

Procedure:

The procedure is a direct continuation of positioning programming. Proceed in accordance with the following:

1. Select the WEAVE function.
2. State which weaving program 1 -9999 is to be used.
3. State the adjustment factor 0.1 -100% for the basic speed of the weaving program.

6.6 REFP

See section 3.9.1 for more detailed description.
and 3.4.1 for parameter DHANDCK.

Program the REFP (reference point) instruction, as follows:

1. Check that the reference point is not active.
2. Program the position (program segment) parallel to desired positions.
3. Program the REFPOINT ON instruction **immediately before** the first position to be displaced.
4. Program the REFPOINT OFF instruction **immediately after** the last position to be displaced.

Note

The reference point is not activated/deactivated until the REFPOINT ON/OFF instruction is executed. Use a FINE position before REFPOINT is activated.

Reference point can be used also with an external axis. In this case you do not achieve a parallel displacement but:

- a rotating external axis moves a number of degrees from its current starting position.
- a linear external axis moves a number of millimeters from its starting position.

6.7 VELCTRL

See section 10.4 for more detailed description.

Speed control is programmed as follows:

- 1 Program a FINE position instruction.
- 2 Press VELCTRL
- 3 Enter the number of the sensor (1-16) with the numerical button set
- 4 Press the function button ENTER. If the question THRESHOLD VALUE=" is presented on the lower line of the display, go directly to point 7. Otherwise go to point 5.
- 5 Specify the speed as a percentage (0-100) of the programmed speed with the numerical button set.
- 6 Press ENTER.
- 7 Specify with the numerical button set the required threshold value as a percentage (0-100) of the maximum signal range of the sensor. If the current signal level is to be a threshold or maximum value, run the robot with the joy stick to the position where the sensor gives the required signal.
- 8 Press ENTER.

The argument is now programmed. If required, further arguments can be added to the instruction.

6.8 CONTOUR

See section 10.5 for more detailed description.

Contour tracing is programmed as follows.

- 1 Program a FINE position instruction.
- 2 Press SCAN
- 3 Press CONTOUR
- 4 Enter the number of the sensor (1-16) with the numerical button set
- 5 Press ENTER. If the question "BIAS =" is shown on the lower line of the display, continue to point 6. Otherwise go to point 8 directly
- 6 Specify with the numerical button set the required bias as a percentage (0-100) of the maximum signal range of the sensor. If the current signal level is to be the bias, run the robot with the joy stick to the position at which the sensor gives the required signal before the value is programmed.
- 7 Press ENTER
- 8 If further sensors are to be used press YES and repeat points 4-6 for the new sensor. Otherwise press NO.

The argument CONTOUR is now programmed. If required, further arguments can be added to the instruction.

Note

This instruction is affected by the use of the speed correction buttons +%, -%

6.9 CORVEC

See section 10.6 for more detailed description.

A correction movement is programmed as follows.

- 1 Check that the relevant sensor is defined.
- 2 Press SCAN twice
- 3 Press COR VEC
- 4 Specify the number of the sensor (1-16) for which the correction movement is to be programmed with the numerical button set
- 5 Press ENTER
- 6 Run the robot with the joystick a short distance in the direction required
- 7 Press END POS

Note

If the operator should release the enabling device during running with the joystick before finishing the sequence the programming of this function has to be started from the beginning.

If the current percent value $V = \dots \%$ on the upper line of the display is to be changed for the correction separately, execute the points 7 and 8 below. Otherwise omit these.

- 8 Specify with the numerical button set the required percentage value for the correction (0.1-799.9)
- 9 Press ENTER.

6.10 POS POS

Position argument for movement

With POS POS a positioning instruction is programmed in such a way that the robot, during program execution, will move to a stored position i. e. with the tool orientation stored. The position can be provided with an offset if required.

1. Select POS POS under the POSITION menu.
2. The system now asks: VIA REG? Answer YES if you wish to specify a register which contains the position register number. Answer NO if you wish to give the position register number directly.
3. Specify a required register number (0 - 199)
4. Specify if a displacement (OFFSET) is to be added. If the answer is NO, the argument is entered and ready.

6 Position instruction menu

5. If the answer is YES, the offset, it can be given in one of the ways described here:

A. Enter the x-, y- and z-values in basic coordinates in mm with one decimal and with the correct sign. If any of the values is to be 0, press ENTER directly. See figure below.

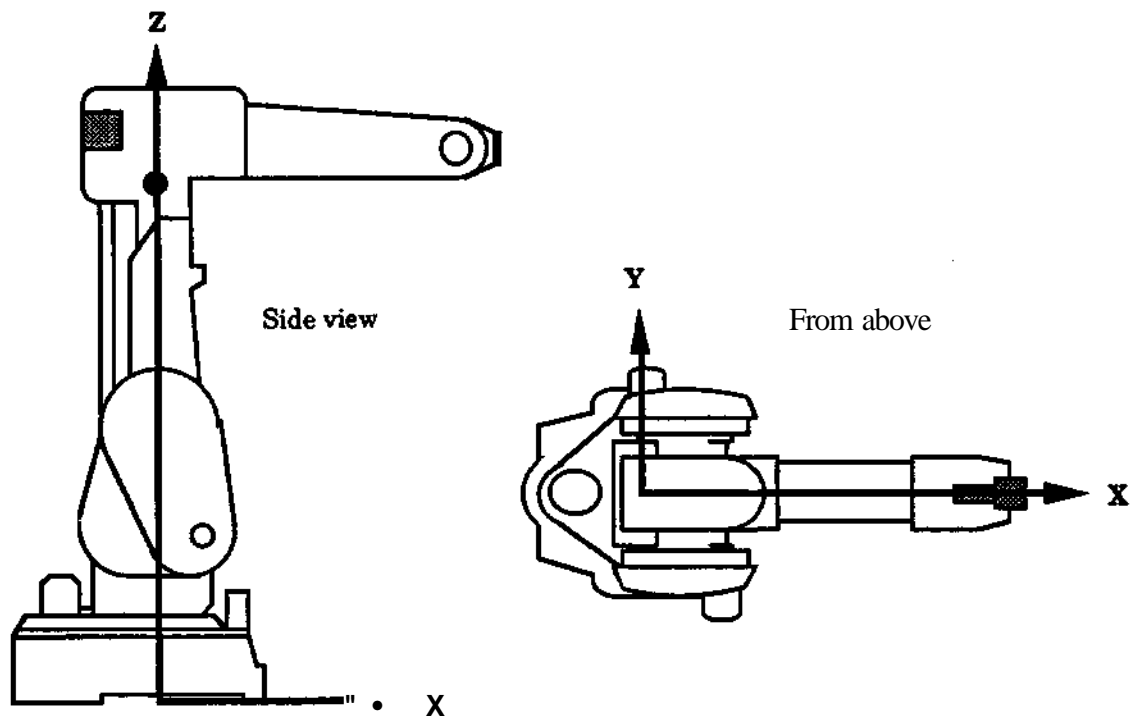
B Run the tool the distance required in the direction required and then press END POS.

C

- Select VIA REG
- Define the scale factor (0.1 - 20, scale factor 1.0 means 1 mm or 0.1 inch).
- Define the data register which contains the required x- displacement.
- Define the data register which contains the required y- displacement.
- Define the data register which contains the required z- displacement.

The latter procedure thus allows entry of displacements from data registers is possible. In this way the offsets can be changed dynamically during program execution.

See section 3.4.1 for more information about, parameter DHANDCK.



6.11 STO POS

Storage of position

Menu: POSITIONING

Function: STO POS

1. Select STO POS under the POSITIONING menu.
2. Specify a position register 0-199 in which the position is to be stored.
3. (Valid if position register 0-49). Specify if the current position of the robot is to be stored at occasion of programming (Answer YES on the question PROGRAMMED POSITION?) or if current position shall be stored during program running (answer NO).

Note that only register 0-95 stores external axes. Register 96-199 does not store external axes.

The instruction is now complete.

Note STO POS is intended for FINE positions. If the instruction is used on other types of position zone, the starting point of the zone (the zone entry point) will be stored.

For more detailed information see section 4.9.2.

If Programmed point is selected, the position will be fixed and can not be changed during execution. I.e. execution of the position does not store the current position of the robot.



(Valid for AW)

Check if reference point shall have influence over stored coordinate values or not. If the reference point shall be counted must the parameter REF POINT MOD be activated.

6.12 POS LOC

Location argument for movement

Menu: POSITIONING

Function: POS LOC

With POS LOC a positioning instruction is programmed in such a way that the robot, during program execution, will move to a stored location i e with the current tool orientation. The location can be provided with an offset if required.

1. Select POS LOC under the POSITIONING menu.
2. The system now asks: VIA REG? Answer with YES if you wish to name the register which contains the number of the position register you wish to use. Answer NO if you wish to give the number of the position register directly.
3. Specify the required register number (0-99)
4. Specify if a displacement (OFFSET) is to be added. If the answer is NO, the argument is entered and ready.
5. The offset, can be given in one of the ways described here:
 - A. Enter the x-, y- and z-values in basic coordinates in mm with one decimal and with the correct sign. If any of the values is to be 0, press ENTER directly.
 - B Run the tool the distance required in the direction required and then press END POS.
 - C
 - Select VIA REG
 - Define the scale factor (0.1 - 20, scale factor 1.0 means 1 mm or 0.1 inch).
 - Define the data register which contains the required x- displacement.
 - Define the data register which contains the required y- displacement.
 - Define the data register which contains the required z- displacement.

The latter procedure thus allows entry of displacements from data registers.

6.13 CIRCLE

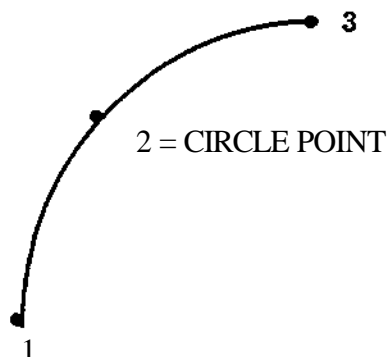
Circular movement

Menu: POSITIONING

Function: CIRCLE

Means:

The tool moves in a circular arc if the program is executing in RECT or MODRECT coordinates.



Facts:

A circular arc < 180 degrees is built up of two standard points and an intermediate circle point. For circular arcs, equal to or greater than 180 degrees, two circular arcs with the same radius must be used in succession (the end point for the first becomes the starting point of the second).

6.14 FRAME

Activation of program displacement

Menu: INSTRUCTION

Function: FRAME

1. Select FRAME under the POSITION menu.
2. Position (1-3) that should be stored.

The new frame will be calculated, when the instruction FRAME (1-5) DEFINE is executed. It will be activated when FRAME (1 - 5) is executed.

For more information, see section 3.9.2, 5.15, 9.6 and 10.7.

See section 3.4.1 for more information about, parameter DHANDCK.

6.15 TIME

Definition of positioning time

Menu: POSITIONING

Function: TIME

1. Select TIME in the menu
2. Specify the positioning time for movement from the preceding position in the program forward to the current position in seconds with one decimal. Max time is 650 seconds.

Note

The TIME-argument can occur together with the following arguments:

SEARCH

POS LOC

FINE C1, C2

RELTOOL (MH/GL/SW)

POS POS

All other arguments:

- Disappear when the TIME function is selected
- Are impermissible when the TIME function has been selected.

6.16 PALLET

(MH/ASM and GLUE)

Means:

A pallet (matrix) is defined by specifying three of the corner points. The robot is then automatically positioned to the different points in the pallet. Every time the PALLET-instruction is executed the positioning will be made to the next element in the pallet.

Facts:

The pallet is defined by specifying the size, number of rows and columns, and by manually moving the robot to three corner points. A position above the pallet can also be defined by giving the fourth position. This position specifies a displacement from the first point. This displacement can then be used for all points to define a fetch position above the pallet. 10 pallets can be defined with a size of up to 99x99 elements in each. Also single columns can be defined.

The pallets can be stored on floppy disc together with the other system data.

The pallets can be placed anywhere in the working area and they need not be rectangular.

Actual position in the pallet is pointed out by two ordinary registers. These are automatically updated when a PALLET-instruction is executed but can also be set by the REGISTER-instruction.

The positioning can be done both to a position above the pallet and to the real pallet position.

When the pallet is defined the orientation of the tool (not the wrist) is stored. This has the effect that a multigrip with different TCPs can be used if the BASEPoints are defined.

Used:

When any type of pallet or matrix-pattern is used.

6 Position instruction menu

Executed:

The robot is positioned to the next position in the pallet. If the last position in the pallet has been passed or if the registers are pointing out a nonexistent position in the pallet an error message is given.

Meny: POSITIONING

Funktion: PALLET

- 1 Select PALLET under the POSITIONING menu.
- 2 Specify pallet number 0-9
- 3 Specify the register which contains actual column.
- 4 Specify the register which contains actual row.
- 5 If positioning shall be done above the pallet select YES, if it shall be done in the palletlevel select NO.

The instruction is now ready.

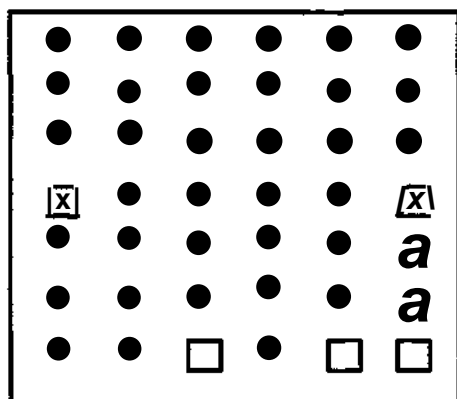
Note!

The registers must be set to correct values before the PALLET instruction is performed.

See section 3.4.1 for more information about, parameter DHANDCK.

Example:

The robot shall fetch details from a pallet and assemble these somewhere. The pallet has been defined as Pallet no. 1. Two points contain no details because they are check holes for the fixture. See figure.



Pallet 1:7 rows - 6 columns

Check hole

PROGRAM 10

i	
i	
100SETR1=1	Initiate
110SETR2=1	Initiate
120SETR3=1	Initiate
130 SET R4 = 1	Initiate
140 JUMP TO 300 IF R1=1 * R2=4	Jump if hole 1
150 JUMP TO 300 IF R1=6 * R2=4	Jump if hole 2
160 POS V=100% PALLET 1 R1 R2 OFFSET	Above pallet
170 POS V=50% FIN PALLET 1R3 R4	In pallet
180 GRIP 1 WATT 0.3 S	Grip
190 POS V=50% RELTOOL DX= - 50	50 mm above pallet
200 CALL PROG11	Assemble
210 JUMP TO 140 IF R2<8	Continue until all details are used.
220 !	Change pallet....
!	
250 JUMP TO 100	
i	
300 SET R1 = R 1+1	Control hole
310SETR3 = R3+1	New column
320 JUMP TO 140 IF R 1 < 7	Jump if not end of row
330 SET R 1 = 1	
340SETR2 = R2+1	New row
350SETR3 = 1	
360 SET R4 = R 4+1	
370 JUMP TO 140	

6.17 RELTOOL

Tool-relative displacement (MH/GL/SW)

Means:

The next goal position is calculated relative to the present position and orientation of the tool and based on the arguments given in the instruction or in the referred registers.

Facts:

The arguments are given as elementary translations and rotations in the rectangular hand or tool coordinate system defined in the chapter 3.2 'Coordinate systems'. If the active TCP does not have a corresponding basepoint (BASEP) the movement is calculated in the hand coordinate system.

The movements (arguments) resemble the joystick movements. No more than six arguments can be given in one instruction.

The translations (DX, DY, DZ) and rotations (RX, RY, RZ using the 'right hand rule') are combined based on the initial orientation of the tool. (For example RX is a rotation about x"-axis.) The resulting position is the result of this combination and the initial position.

Additional POSITIONING arguments (V(%), FINE, TIME, SEARCH, CONTOUR and VELCTRL) can be given with RELTOOL.

The translational arguments are given in mm or inches and the rotational in degrees.

6 Position instruction menu

When the arguments are given in registers, they can be scaled with a scale factor, that can have values from 0.1 to 20 with 0.1 increments. Then the real argument value is the integer in the referred register multiplied with the scale factor. The resolution of the values in the registers is 1 mm, 0.1 inch and 1 degree with scale factor of 1.

For safety reasons limit values for the arguments can not be exceeded. They are 400 mm for translations and 180 degrees for rotations.

The arguments can be modified by the DISPL-function.

Used:

When a movement relative to the present position and orientation of tool is desired. The arguments can be also calculated outside robot and transferred to robot via a port as described in the section 5.8.

Executed:

As a special variant of POSITIONING. The movement is performed if the arguments are within the safety limits.

The movements are always performed in the rectangular coordinates (RECT).

6.17.1 REL TOOL (X.Y.Z)

Procedure:

POS RELTOOL with immediate arguments

Menu: POSITIONING

Function: RELTOOL

1. Select RELTOOL under the POSITIONING menu.
2. Specify that the arguments are given in the instruction by pressing NO.
3. Enter translational arguments (DX, DY, DZ) as required
4. Enter rotational arguments (RX, RY, RZ) as required.
5. Push 'BREAK.
6. Enter other POSITIONING arguments.

The instruction is now complete.

Recommendation:

The instruction should be programmed with FINE point.

6.17.2 REL TOOL (via REG)

Procedure:

POS RELTOOL with pointers to registers containing arguments

Menu: POSITIONING

FunctionrRELTOOL

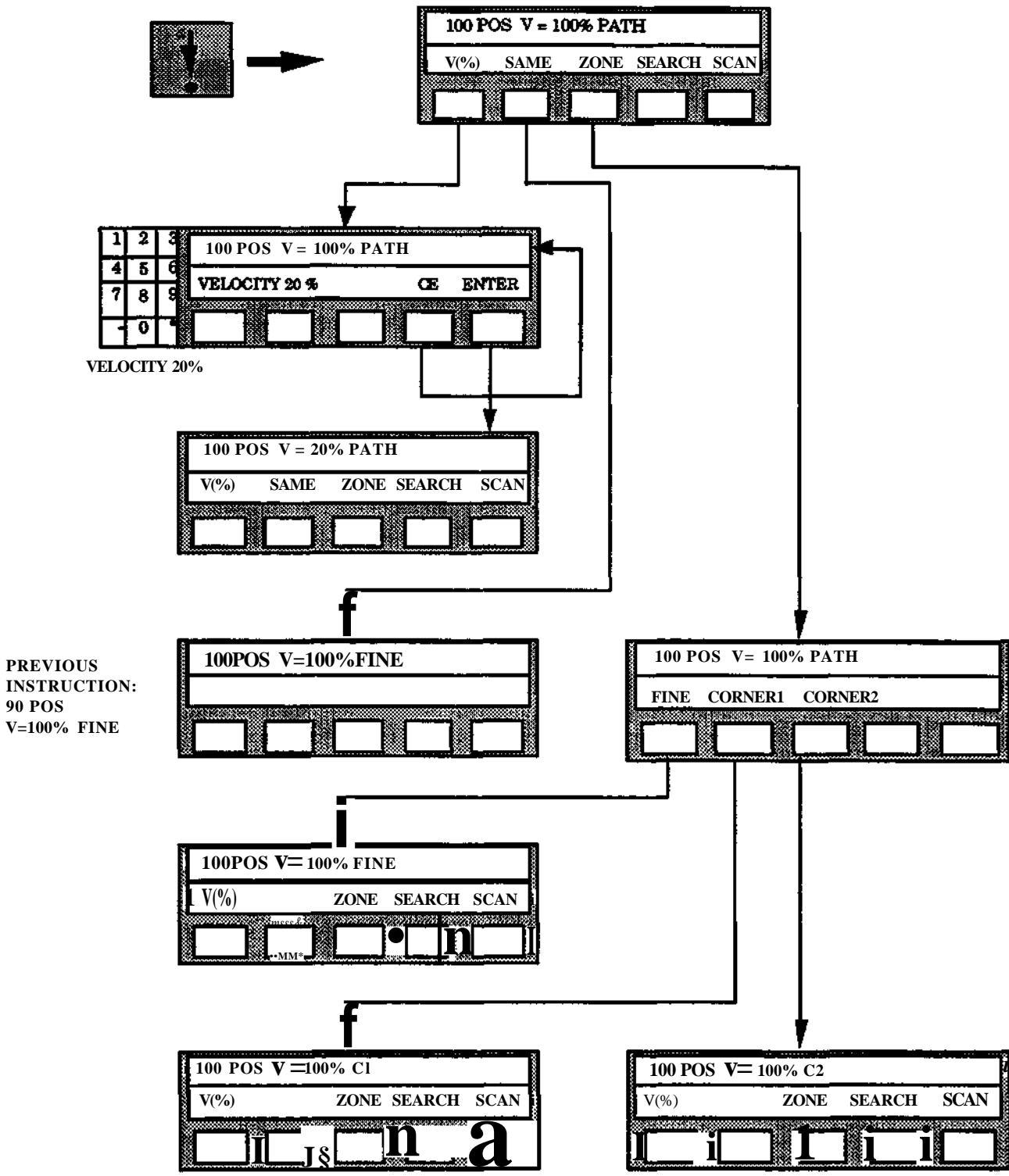
1. Select RELTOOL under the POSITIONING menu.
2. Specify that the arguments are given in registers by pressing YES.
3. Enter the scaling factor for the arguments.
4. Enter register numbers containing translational arguments (DX, DY, DZ) as required.
5. Enter registers numbers containing rotational arguments (RX, RY, RZ) as required
6. Push'BREAK.
7. Enter other POSITIONING arguments.

The instruction is now complete.

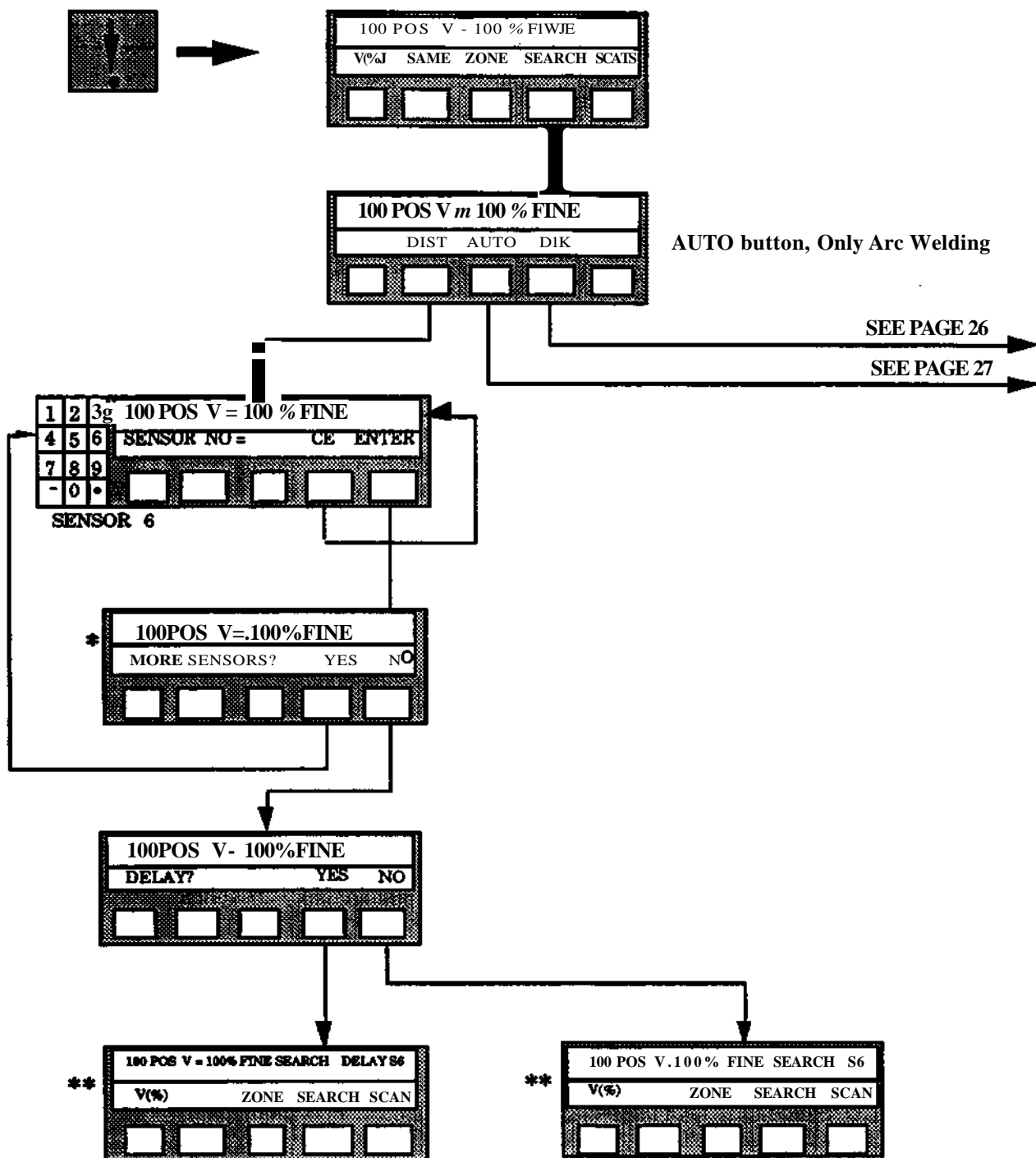
6.18EXTFRAME (AW only)

Programming of EXTFRAME, see section 12.6.5.

6 Position instruction menu



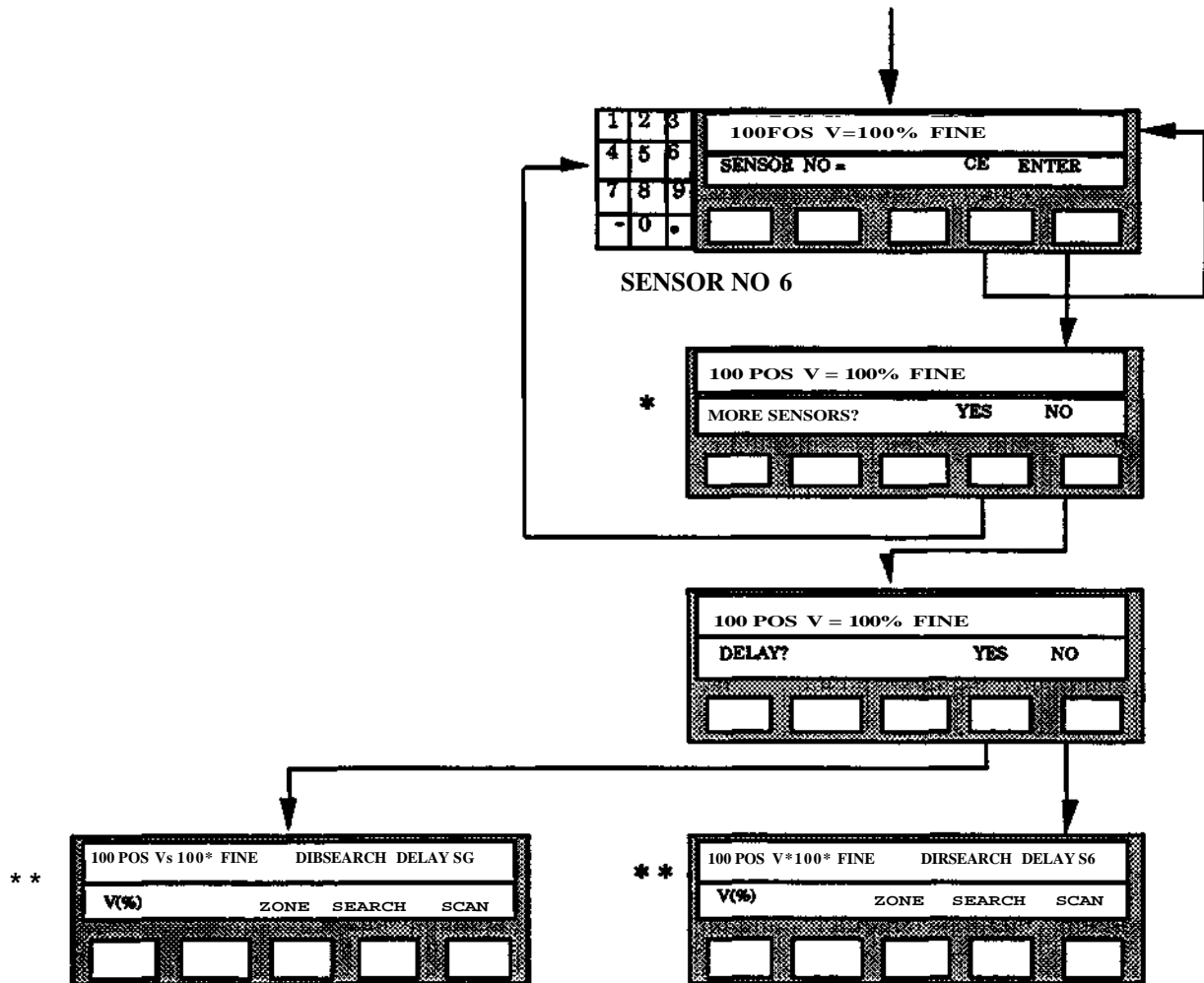
6 Position instruction menu



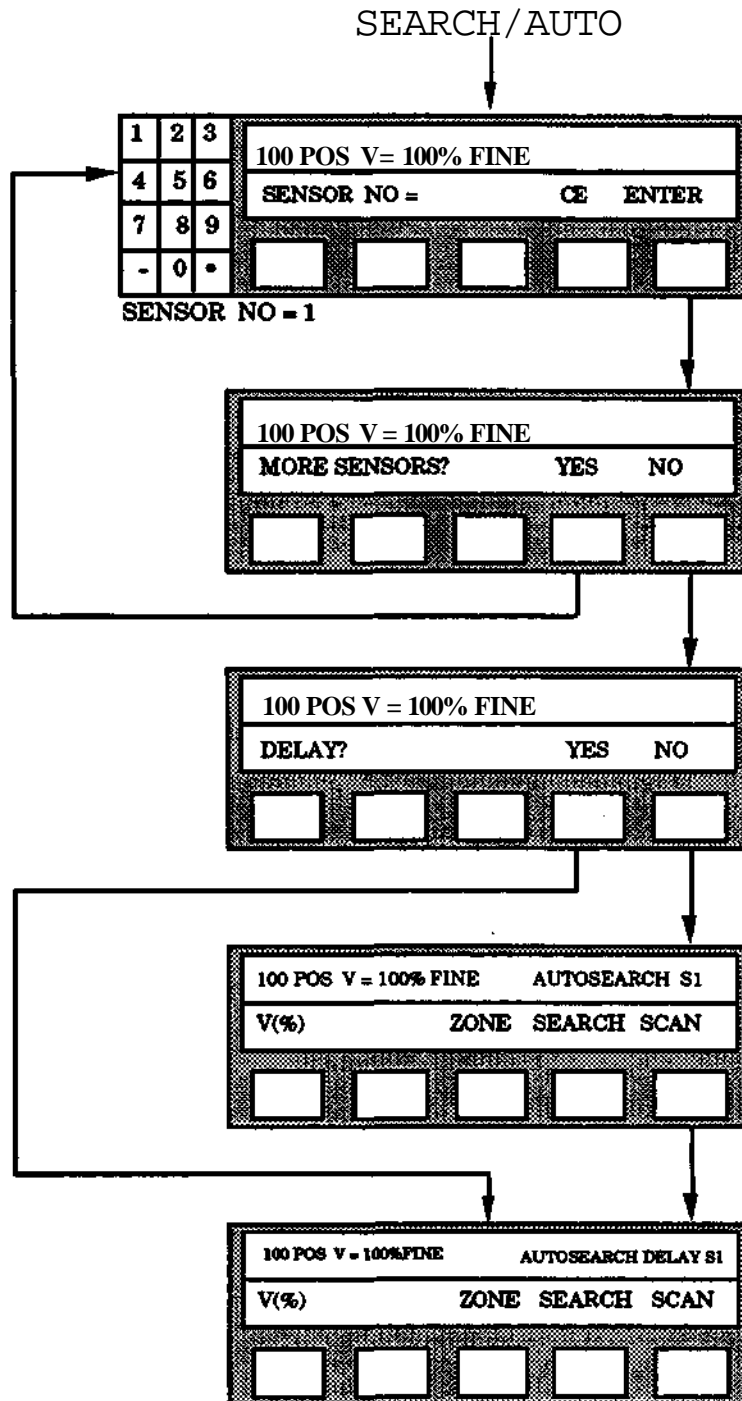
- * If an analog or digital multibitsensor is used, the system asks for a stop condition before this is shown on the display. The answer is to be entered in percent of the maximum sensor voltage.
- ** The stop condition for the current multibitsensor is displayed when the instruction is completed (e.g. S6/45%).

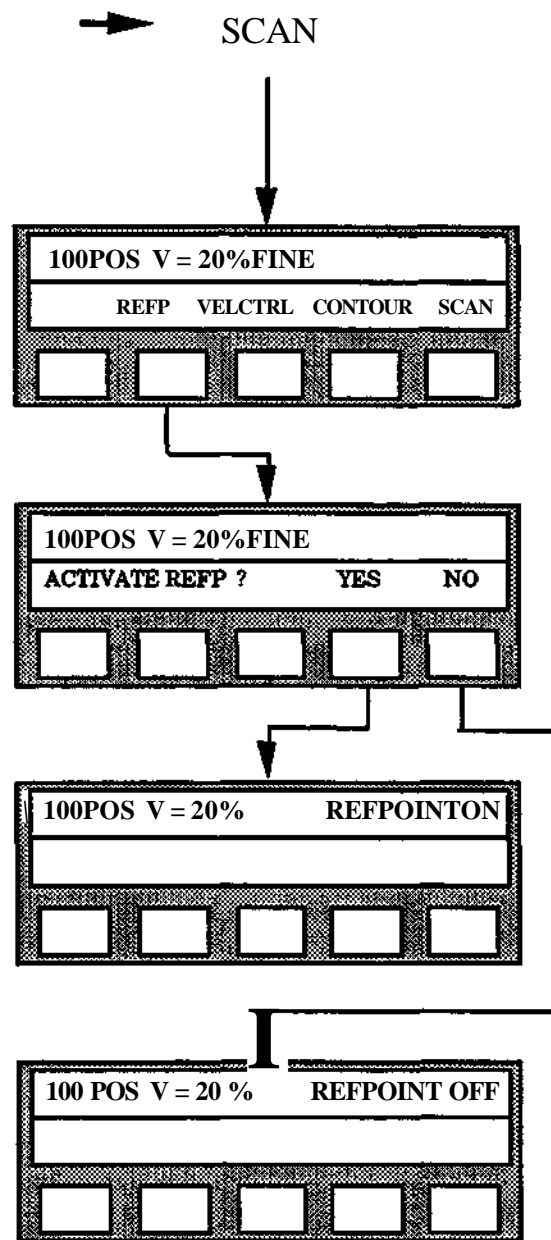
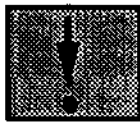
6 Position instruction menu

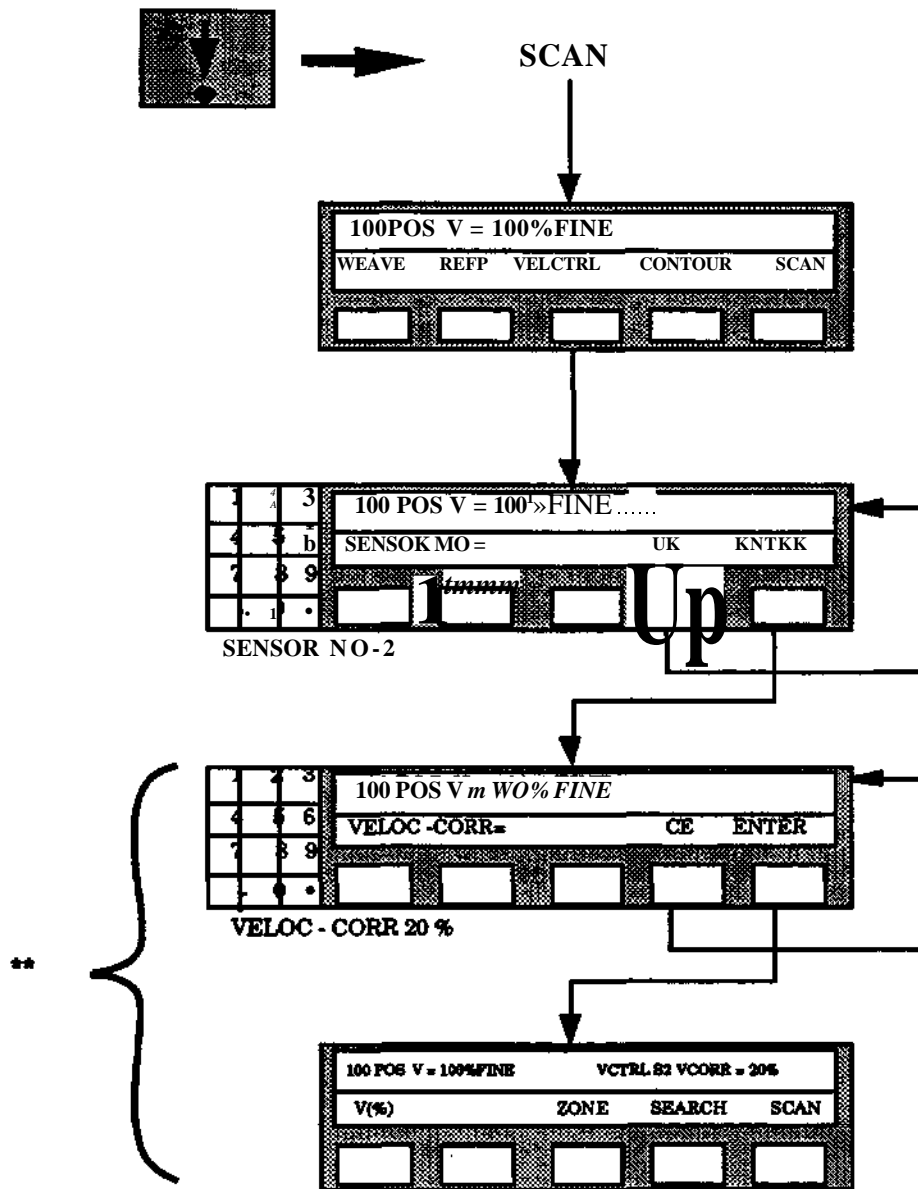
SEARCH/DIR



- * If an analog or digital multibitsensor is used, the system asks for a stop condition before this is shown on the display. The answer is to be entered in percent of the **maximum** sensor voltage.
- ** The stop condition for the current multibitsensor is displayed when the instruction is completed (e.g. S6/45%)

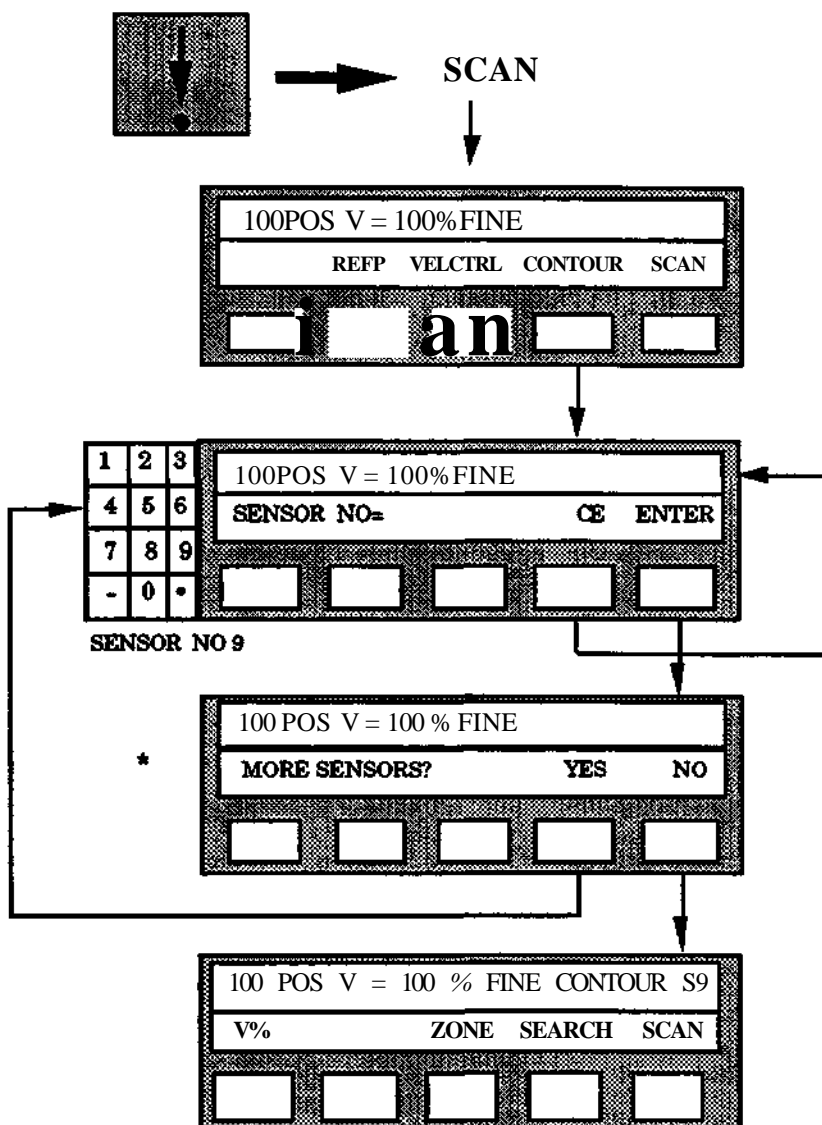




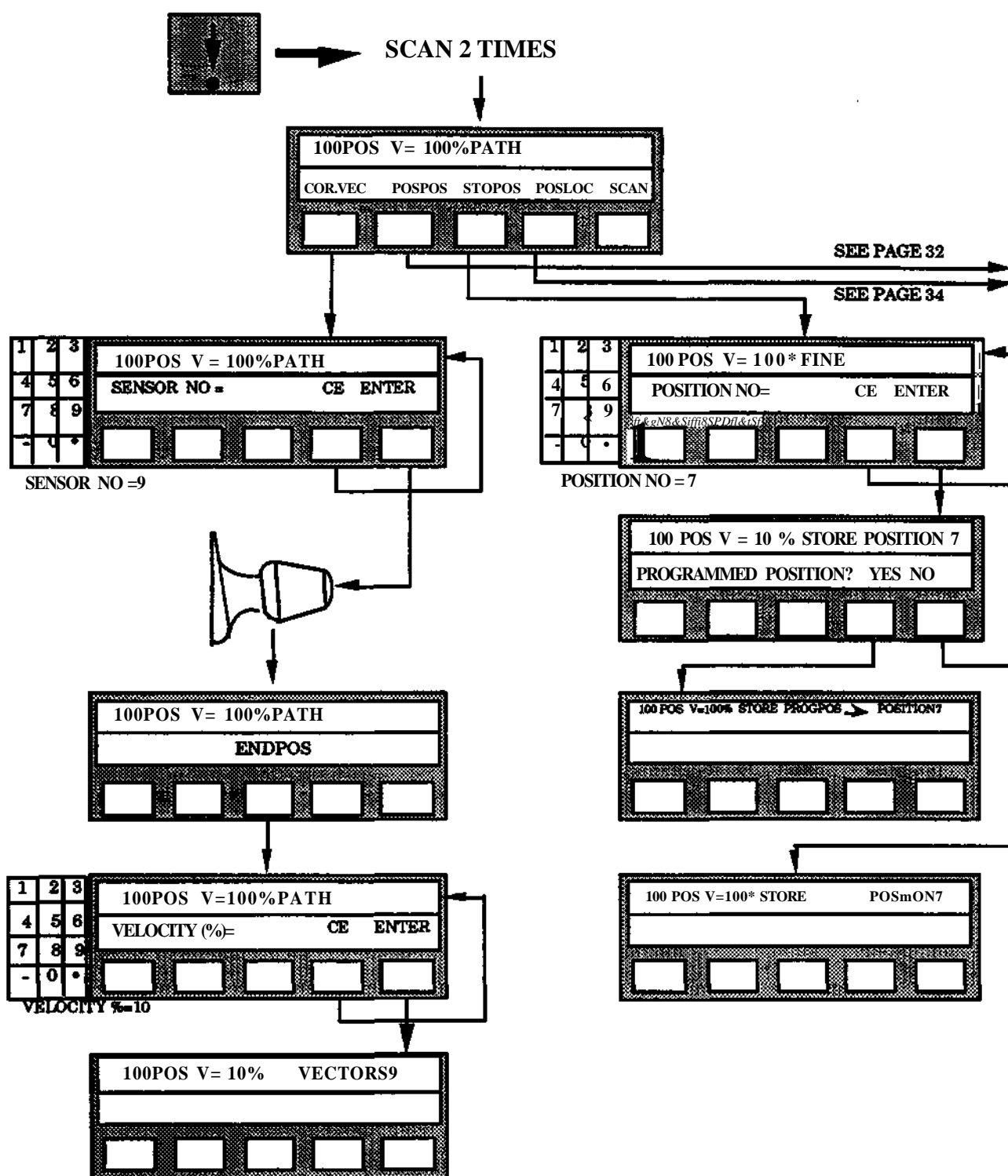


** For multibit sensors these texts are replaced according to following:

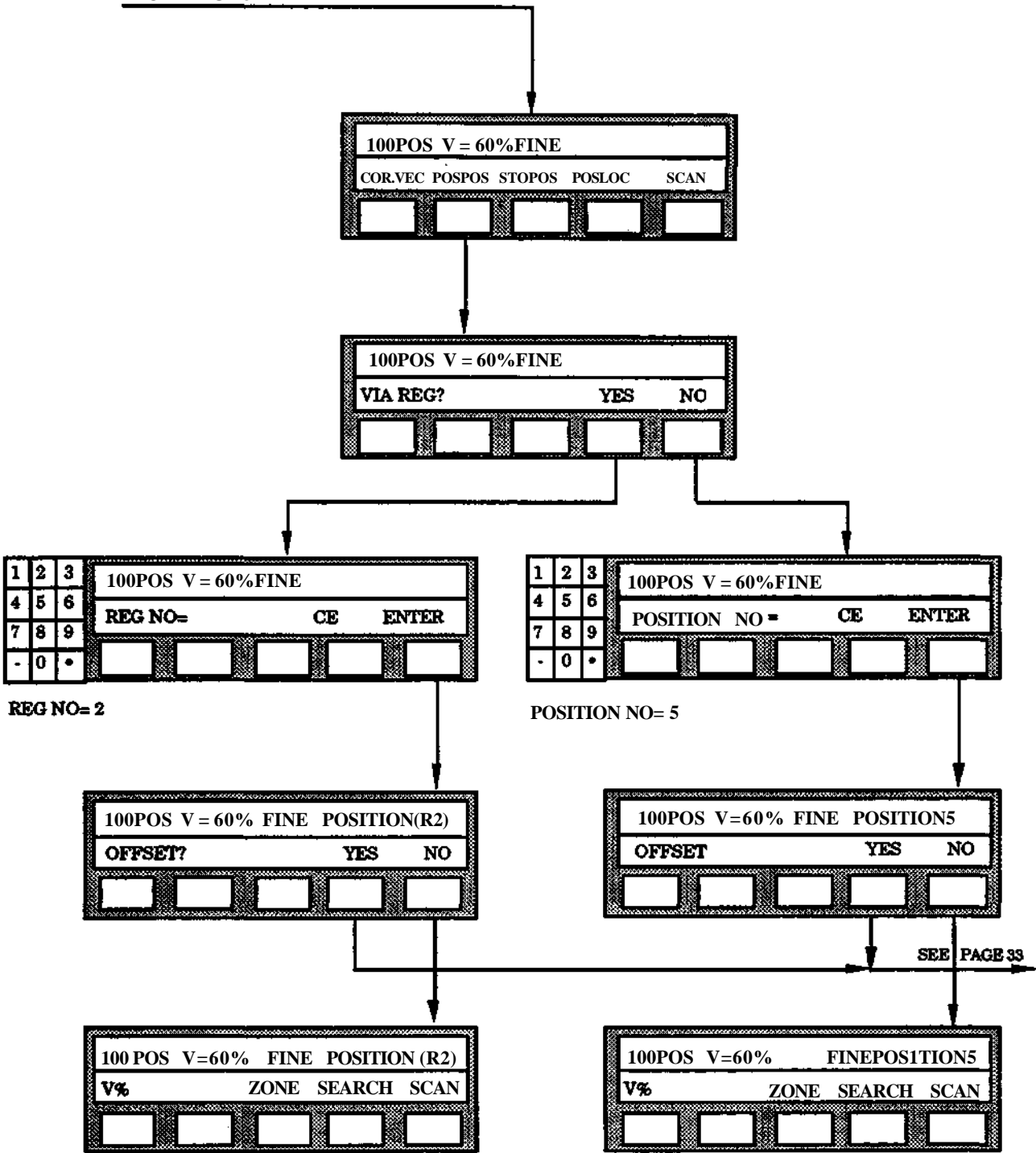
- The threshold value, before the speed is to decrease, is specified in % of the maximum sensor voltage.
- * The argument to the positioning instruction is presented without "VCORR = 20%" instead threshold and maximum values are displayed (e.g. S2/20% max = 75%)



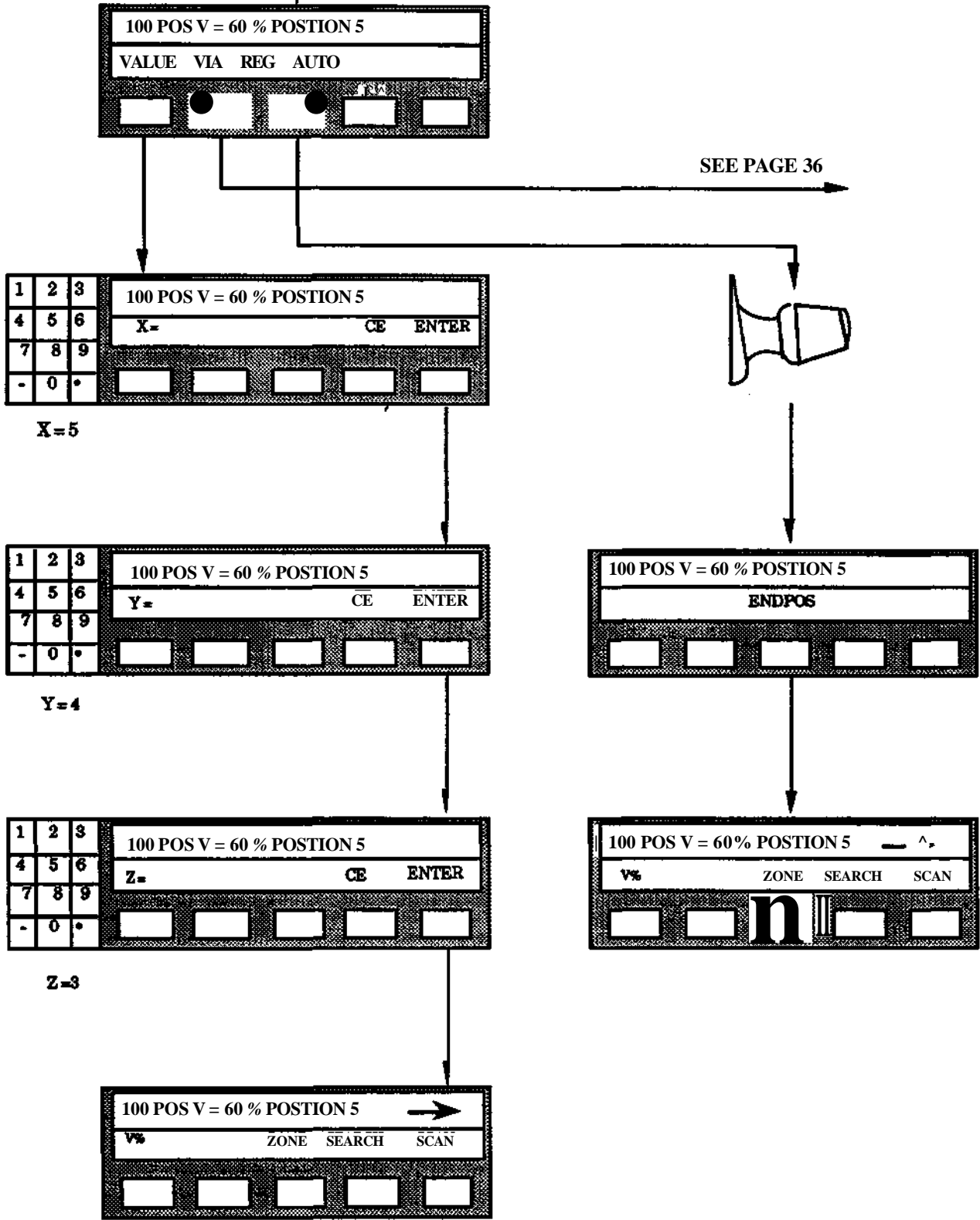
- * If analog or digital multibitsensor is used, the system asks for a bias before this is shown on the display. The answer is to be entered in percent of the maximum sensor voltage. It is displayed when the instruction is completed (e.g. S9/40%)



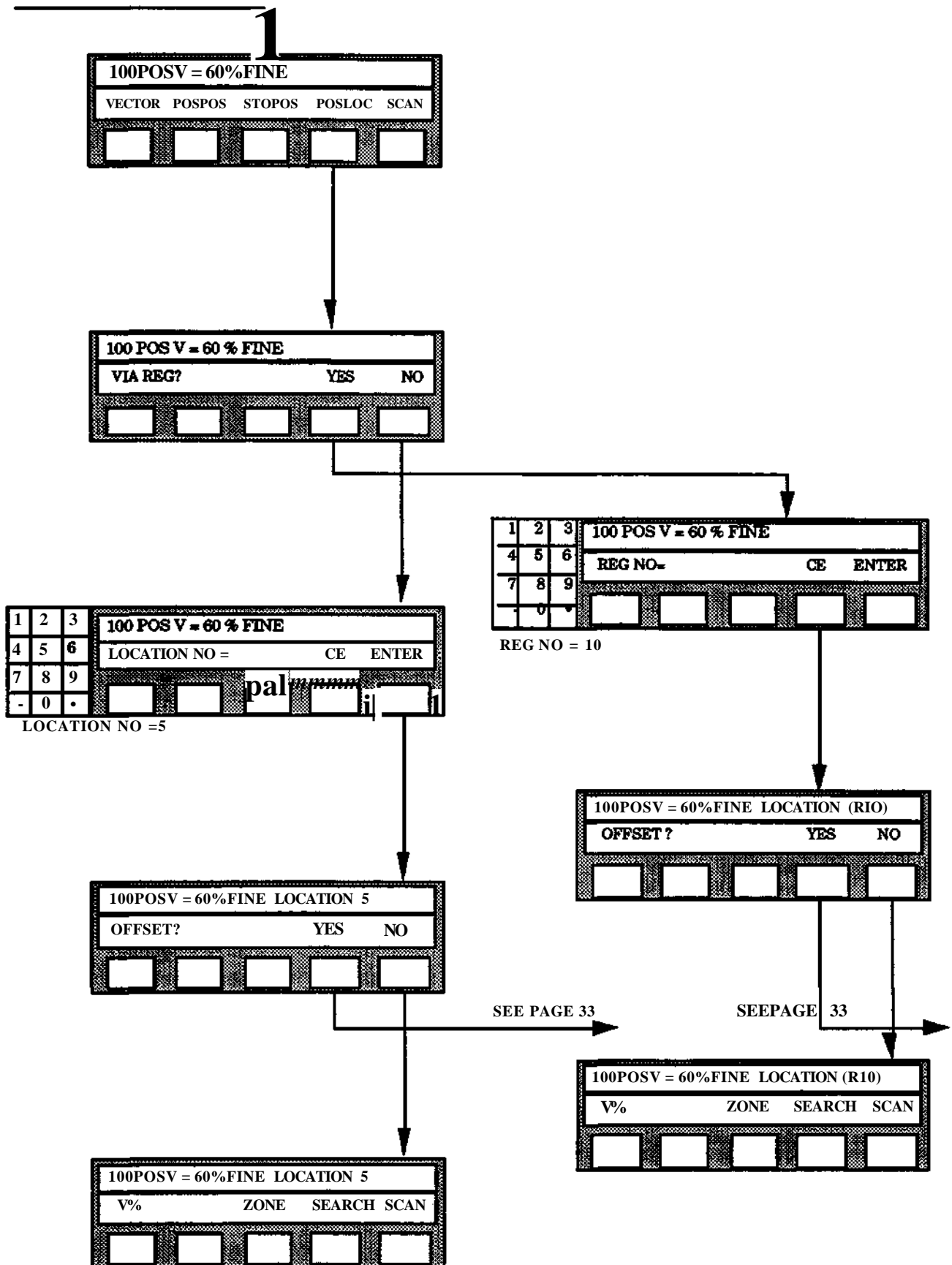
FROM PAGE 31



FROM PAGE 31

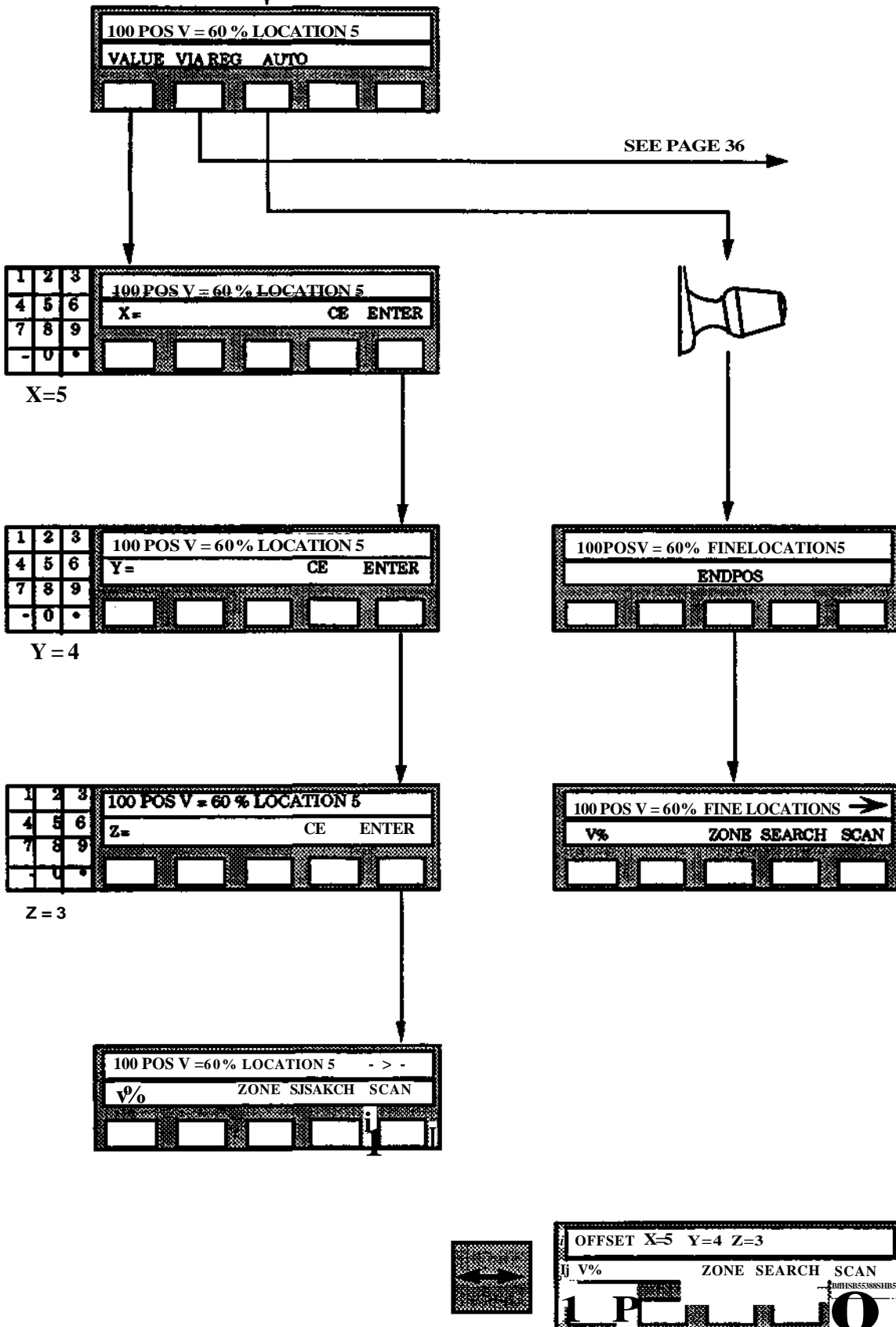


FROM PAGE 31

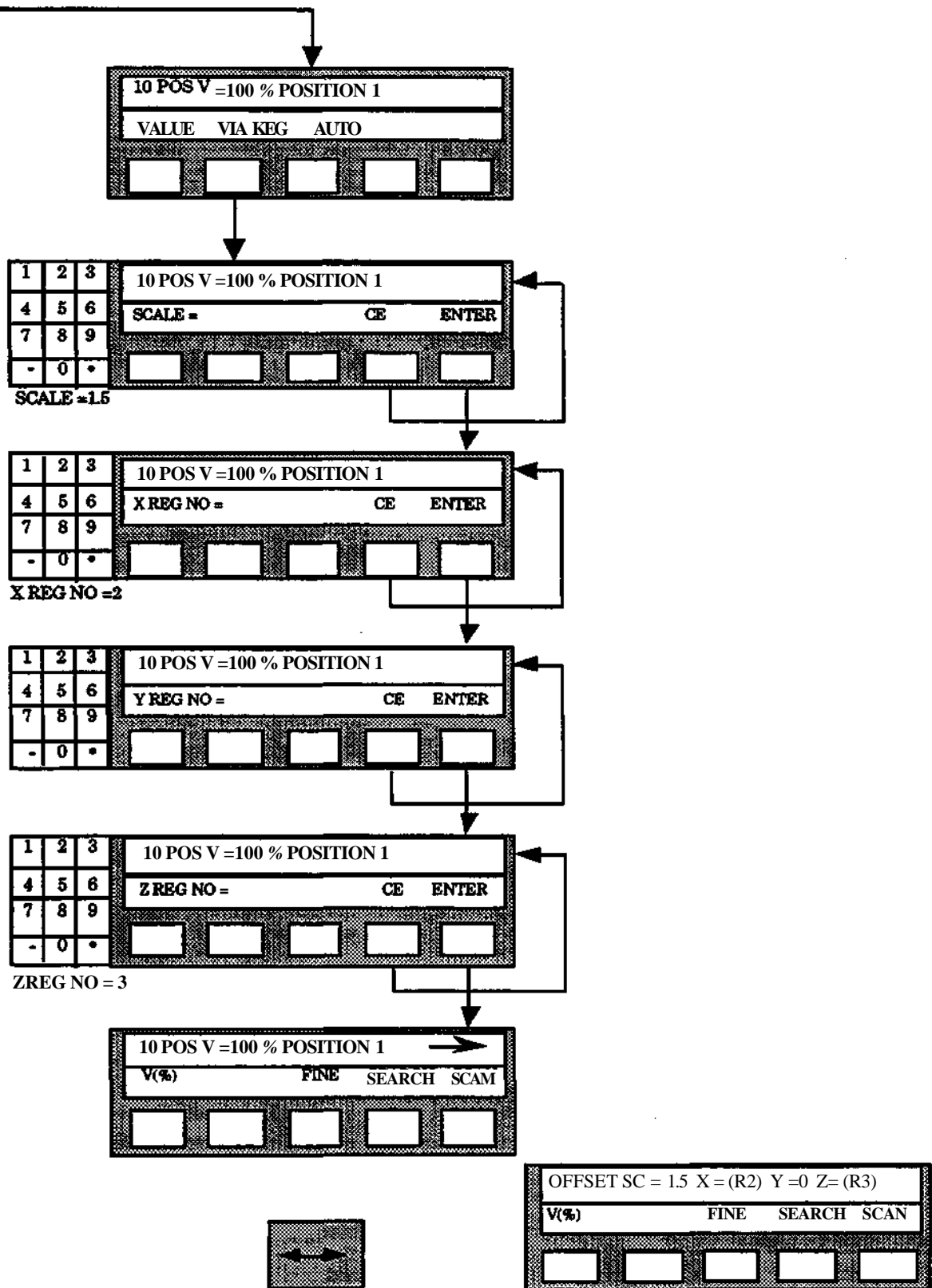


6 Position instruction menu

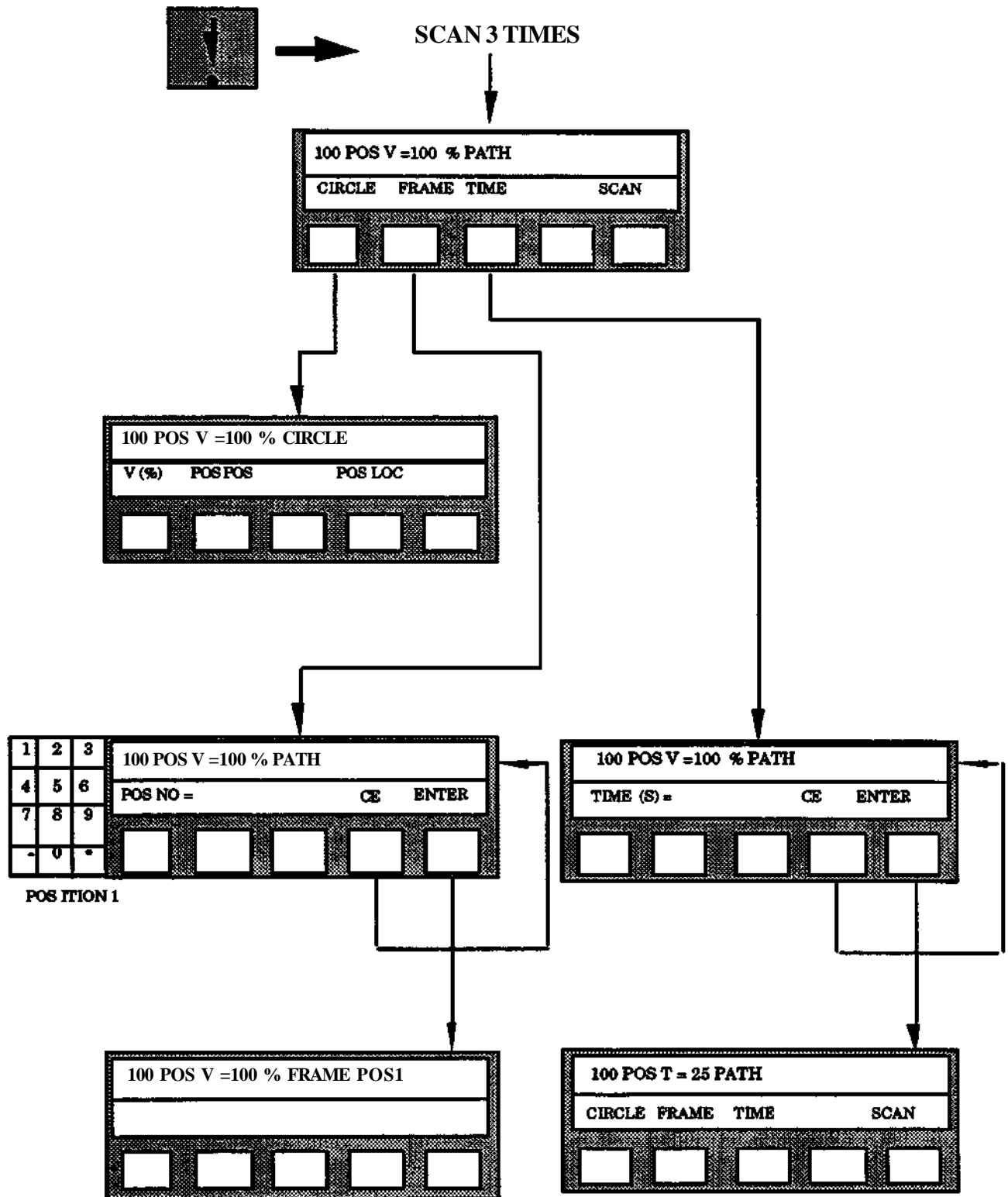
PROM PAGE 31



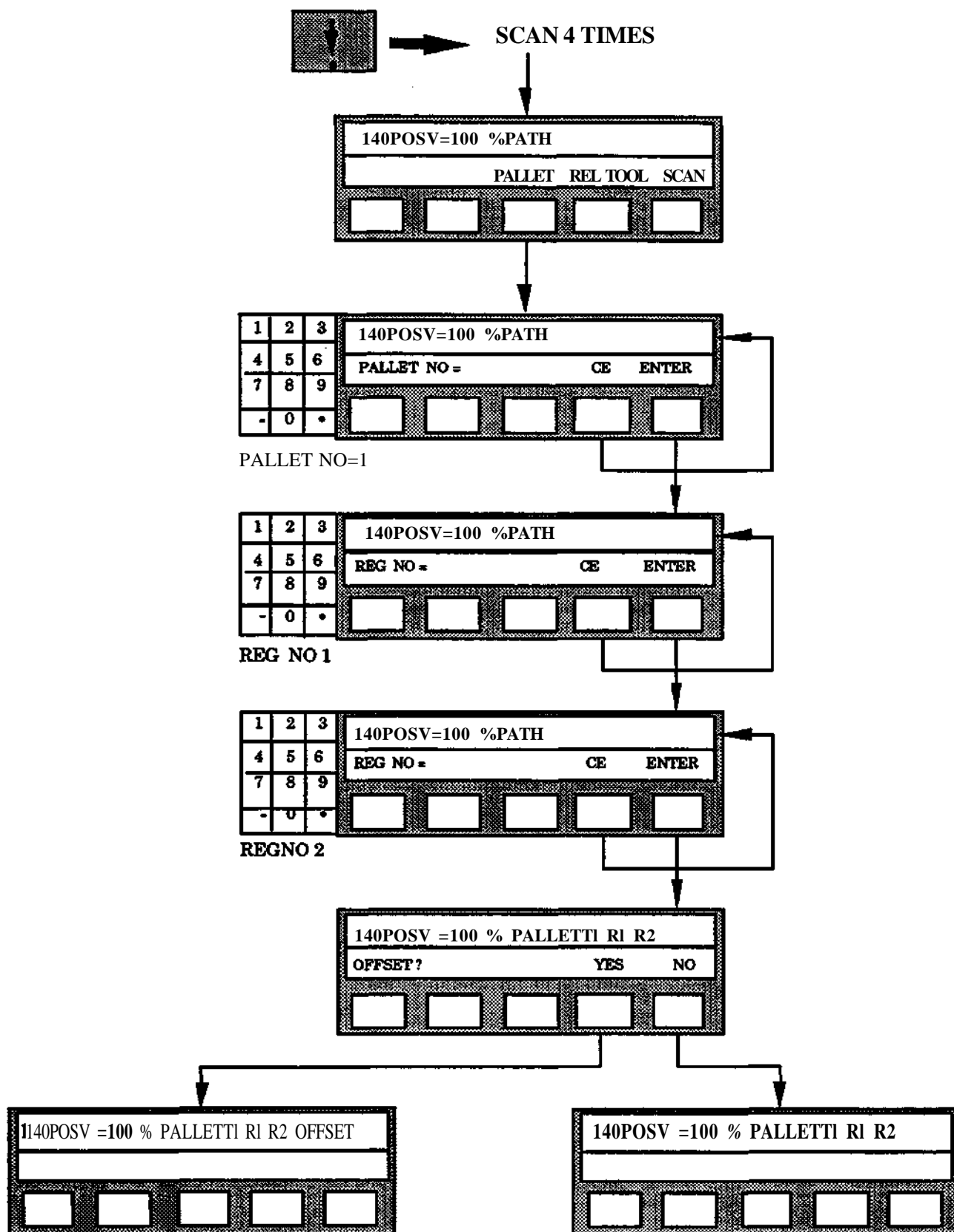
FROM PAGE 33 and 35



6 Position instruction menu



6 Position instruction menu



Automatic menu

Section	Page
7.1-4 PROGST., INS ST., BWD., SIM	7:5
7.5 DISPL	7:5
7.6 CORV ST	7:6
7.7 ALIGN	7:6
7.7.1 Tool direction	
7.7.2 External axes	
7.7.3 Homeposition	
7.8 AW REST	7:8
7.9 RESYNC	7:8
7.10 MOV REST	7:9

7 Automatic menu

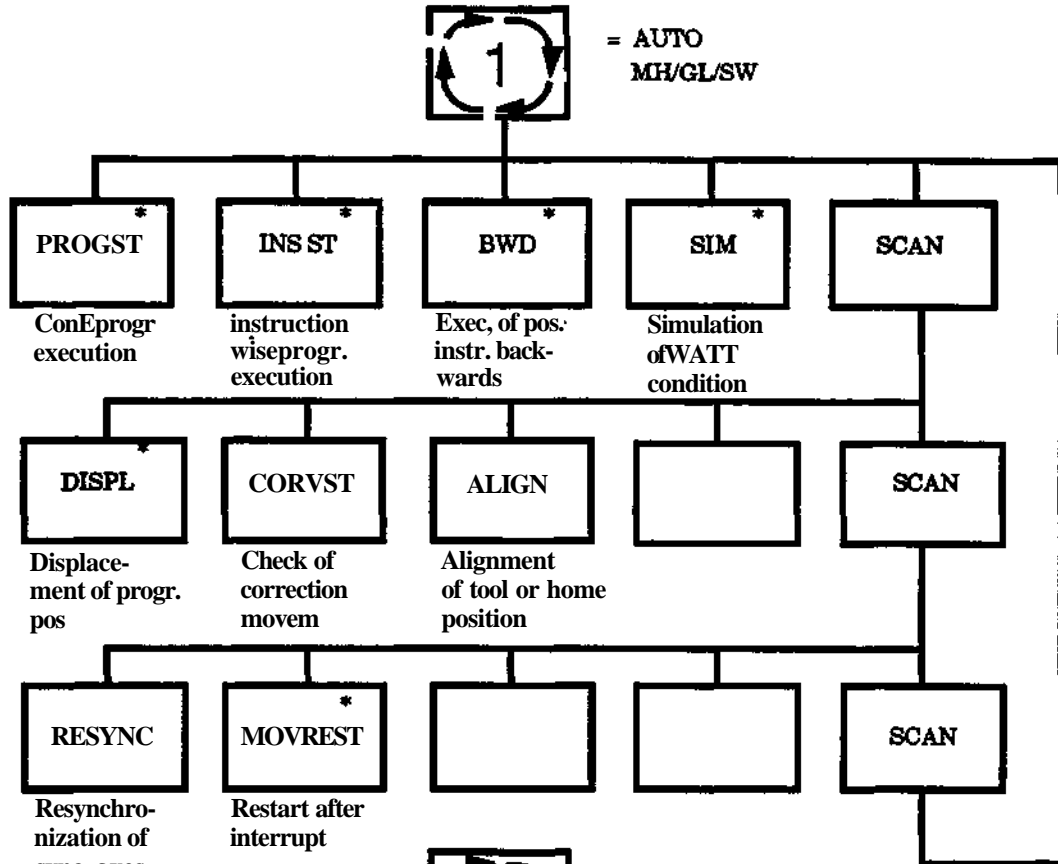
✓

✓

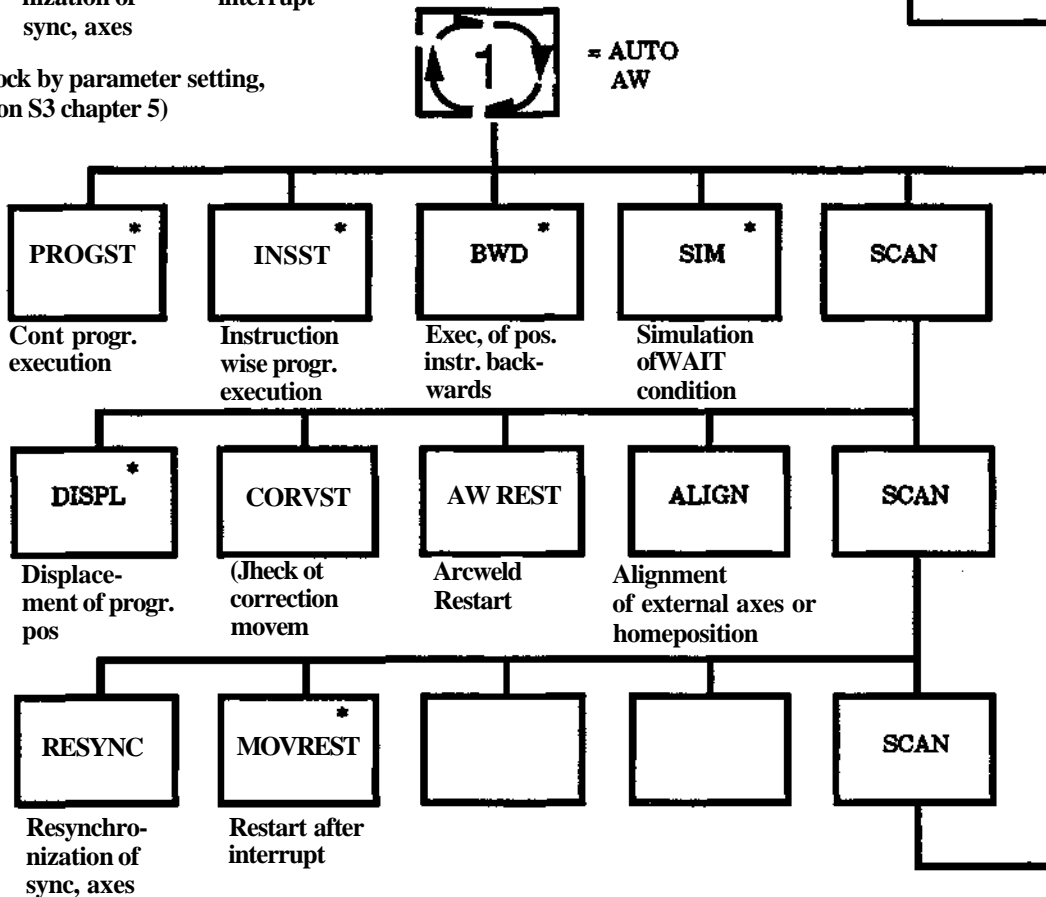
✓

✓

7 Automatic menu



* Possible to block by parameter setting,
(see Installation S3 chapter 5)



7 Automatic menu

7.1-4

PROG ST., INS ST., BWD., SIM

Menu: AUTOMATIC	Functions: see below
PROGST	Continuous execution forward.
INSST	Execution instruction by instruction forward.
BWD	Execution of tool movements only, instruction by instruction backwards, logical instructions must be skipped over using the SKIP BW button. Position instructions can be skipped if desired.
	The PROG ST, INS ST resp. EXEC BW keys must be pressed during program running in manual full speed mode. When execution in MANUAL REDUCED SPEED mode it is possible to disconnect this function with the parameter HOLD RC.
SIM	Simulation of input conditions for WAIT instruction. If the button is depressed when a WAIT instruction is executed, all conditions are assumed to be satisfied.



Program start (PROG ST, INS ST, EXEC BW, AW REST) means that the robot will start moving at high speed. Make sure nobody remains inside the robot working range before starting the robot and make sure all safety equipment is functioning correctly.

7.5 DISPL

Definition of position by reading in during program execution.

Menu: AUTOMATIC Function: DISPL

1. Select DISPL during programming program execution.
2. Specify the program 0- 9999 within which the position to be changed is located.
3. Specify the instruction number for the position within the program 10 - 65 535.
4. Specify the displacement from the present position in mm, with one decimal, for the X-, Y- and Z-coordinates in the base coordinate system. If any of the displacements are zero press ENTER directly when the coordinate value concerned is requested by the system. Permitted maximum value for the different coordinate displacement is 10 mm.

Interrupt the procedure completely by selecting the AUTOMATIC menu again. Information about the active program etc. is then presented again.

Select the AUTOMATIC menu again if the displacement is to be executed again with this function by:

- A position which is before the newly adjusted position in the same program.
- A position within another program.

7.6 CORVST

Execution of correction movement

Menu: AUTOMATIC

Function: CORV ST

The CORV ST function is used when the correction movement for a specific sensor is to be checked. The check is performed according to the following:

1. Call the vector instruction required, see program selection and searching within the program.
2. Press AUTO
3. Press SCAN
4. Press CORVST

When point 4 as above is performed, the robot will move in the direction programmed for the activated sensor (Activation takes place when the desired instruction is executed).

Note. If the sensor concerned is mounted on the robot in accordance with the sensor data entered (see chapter 9.5), the direction of the robot orientation is affected when the button CORV ST is pressed.

To enable the function the robot must be in AUTO with no program execution.

7.7 ALIGN

7.7.1 Tool direction (MH/GL/SW only)

Menu: AUTOMATIC

Function: ALIGN

Means:

The tool is rotated to desired orientation while TCP stand still. With BASE mode the x"-axis of tool (see chapter 3.2.3) is aligned with one axis of the rectangular basic coordinate system.

Facts:

Ten different orientations of (0-9) can be stored and used afterwards. For practical reasons the tool should be rotated at first with a joystick close to the desired orientation. If the overall rotation will be wider than 20 degrees in the BASE mode and 45 degrees in the FETCH mode, an error message is displayed.

Because ALIGN depends excessively on the definition of the TCP, it should be checked before ALIGN is used at first time with a TCP.

The TCP can be changed between storing and reusing an orientation.

Stored orientations are transferred to and from diskette with other system parameters.

Used:

When a specific, or repeated orientation is used in a robot program. Thus it is used during teaching of positions.

7 Automatic menu

Executed:

Only if the desired rotation is not wider than the limits above and the operator starts the movement.

Procedures:

Storing an orientation (STORE)

1. Select ALIGN under the AUTOMATIC menu.
2. Select STORE.
3. Enter the number of the orientation.

Reusing a stored orientation (FETCH)

1. Select ALIGN under the AUTOMATIC menu.
2. Select FETCH.
3. Enter the number of the orientation.
4. Start align movement.

The robot will move with slow speed and "ROBOT ALIGNED" will be displayed.

Aligning the tool to a base coordinate system axis. (BASE)

1. Select ALIGN under the AUTOMATIC menu.
 2. Select BASE.
 3. Start align movement.
- The robot will move with slow speed and "ROBOT ALIGNED" will be displayed.

7.7.2

External axes, (AW only)

Menu: AUTOMATIC

Function: ALIGN

Alignment of external axes together with the EXTFRAME function.

See description of EXTFRAME in section 12.6.5.

1. Select ALIGN under the AUTOMATIC menu.
2. Enter the EXTFRAME number
3. Start align movement

The external axes will move with slow speed and "EXTERNAL AXES ALIGNED" will be displayed.

7.7.3 HOMEPOS

Menu: AUTOMATIC

Function: ALIGN

Alignment of robot axes towards a defined home position.

See description of HOMEPOS in section 9.17

1. Select ALIGN under the AUTOMATIC menu.
2. Select HOMEPO under ALIGN.
3. Select homapos number and start align.

The robot will move in robot coordinates with slow speed to the specified home position.

7.8 AW REST (AW only)

Arcweld restart

The function AW REST facilitates restarting of the welding process after an interruption. To make the welding seam as good as possible, the robot is moved to an optional position which it had 0 - 0.45 seconds before the interruption.

(See Installation S3 for parameter setting). Then the welding process is restarted with the same welding parameters as were used before the interruption. AW restart is performed as follows:

1. Press AUTO.
2. Press SCAN.
3. Press AW REST.

Note.

A restart attempt when it is not allowed will result in an error message: "NOT ALLOWED COMMAND".



Program start (PROG ST, INS ST, EXEC BW, AW BEST) means that the robot will start moving at highspeed.

Make sure nobody remains inside the robot working range before starting the robot and make sure all safety equipment is functioning correctly.

7.9 RESYNC

Using the function RESYNC it is possible to make synchronized axes unsynchronized without reinitiating the entire system. This is done in accordance with the following:

1. Press AUTO.
2. Press SCAN twice.
3. Press RESYNC.
4. If the robot includes synchronized external axes then press SYNC while will commence the synchronization movement.
5. Calibrate the robot. (See Service manual IRB XXXX.)

7.10 MOV REST

The function Move Restart is used for restarting the robot when there is a risk for collisions, e.g. after an emergency stop (see 3.6.1.5).

The robot starts from standing still (2) by moving back (3) to the path, to approximately where the interrupt occurred (1).

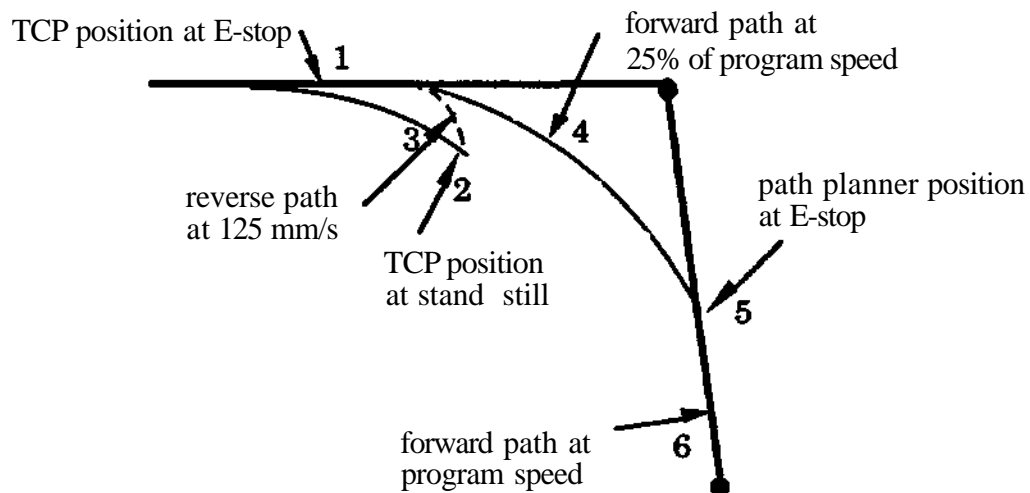
This movement is made at about 125 mm/s.

The robot then moves forward (4) on the path at a speed of 25% of program speed.

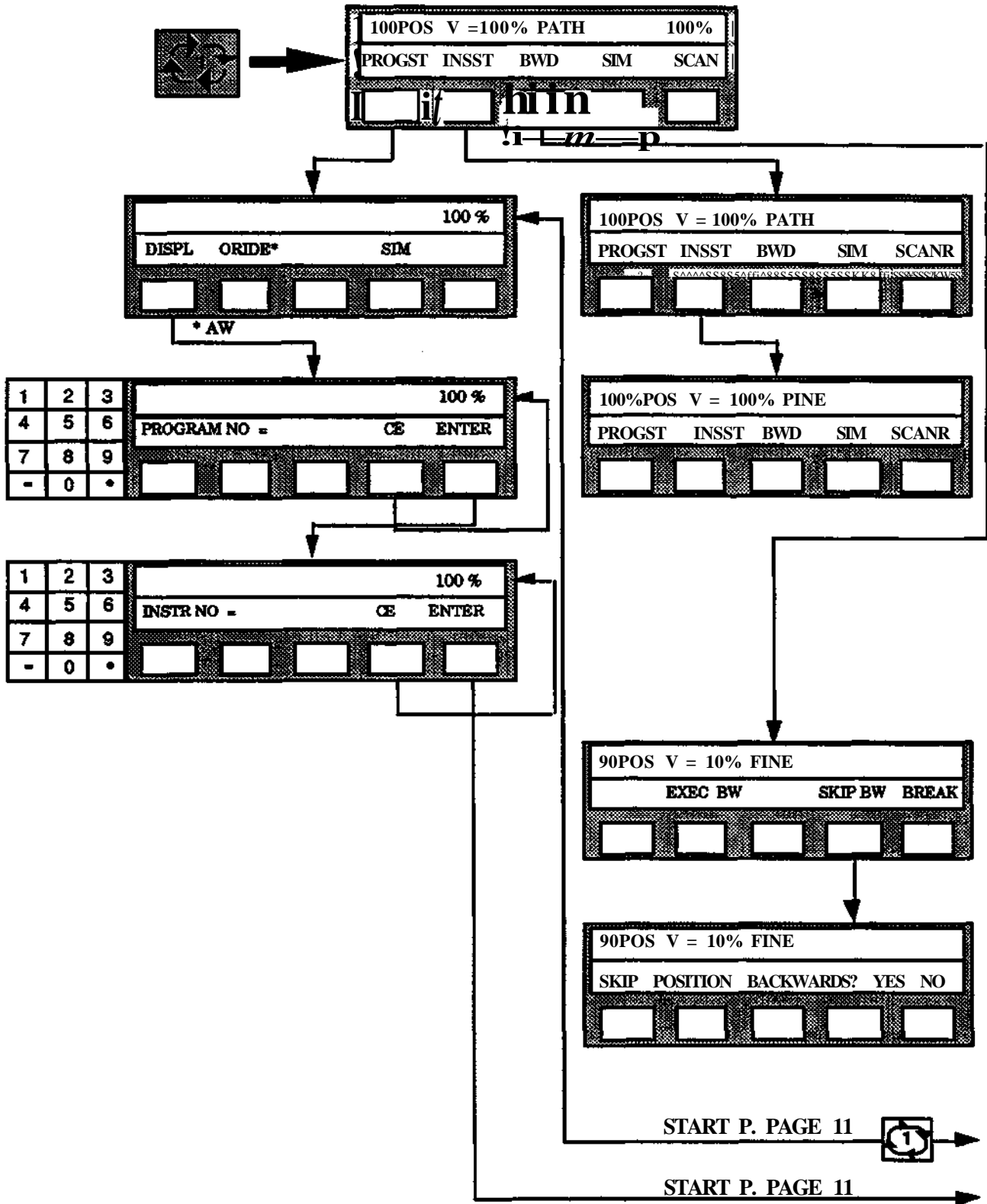
When the TCP has reached the position (5) of the path planner at the interrupt, normal program execution is resumed (6).

All positioning instructions except WEAVE, PALLET and RELTOOL can be restarted.

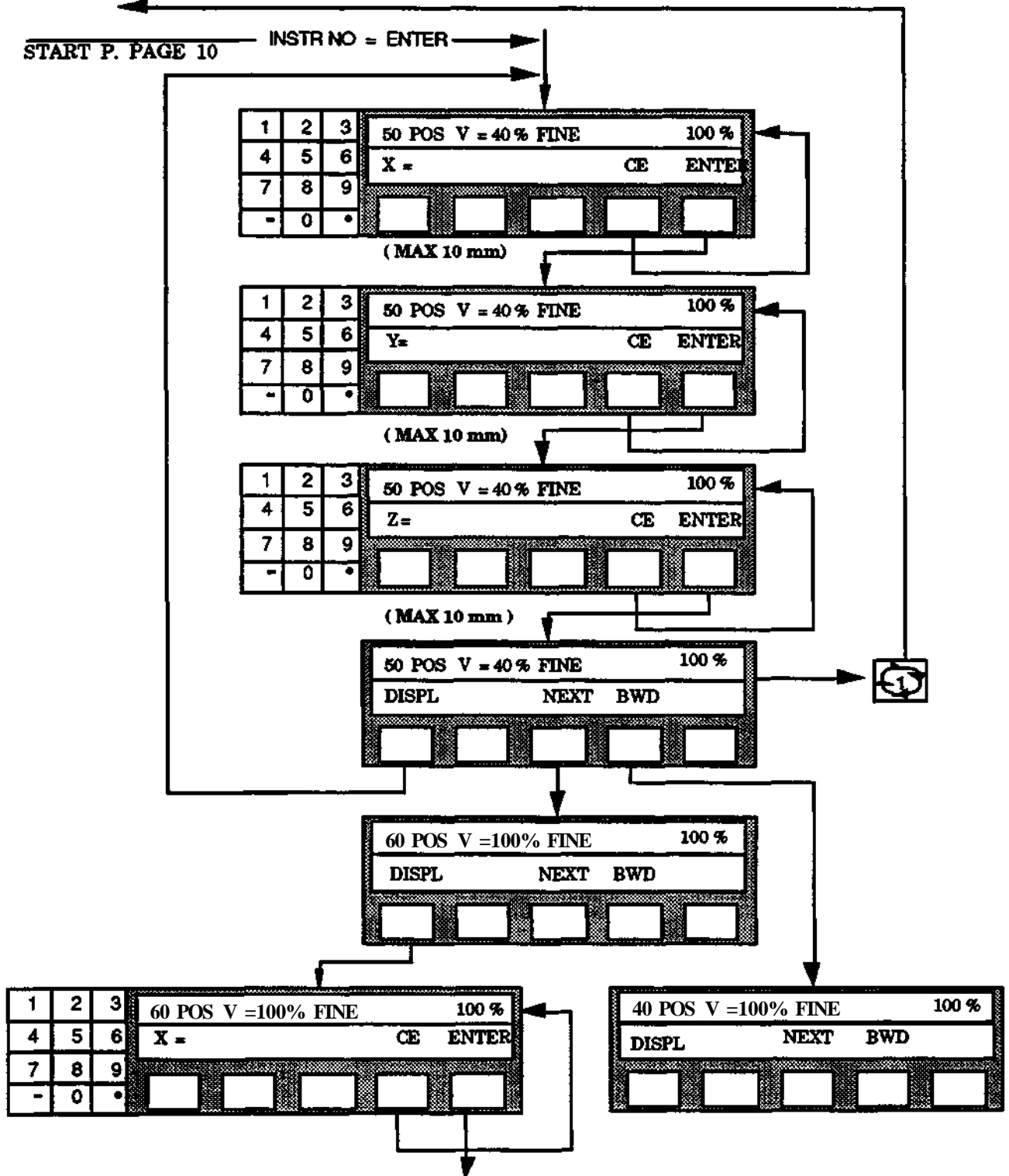
Note that a continuous process, like gluing and arc welding is not resumed until the next process instruction is executed (for AW; use AW restart).

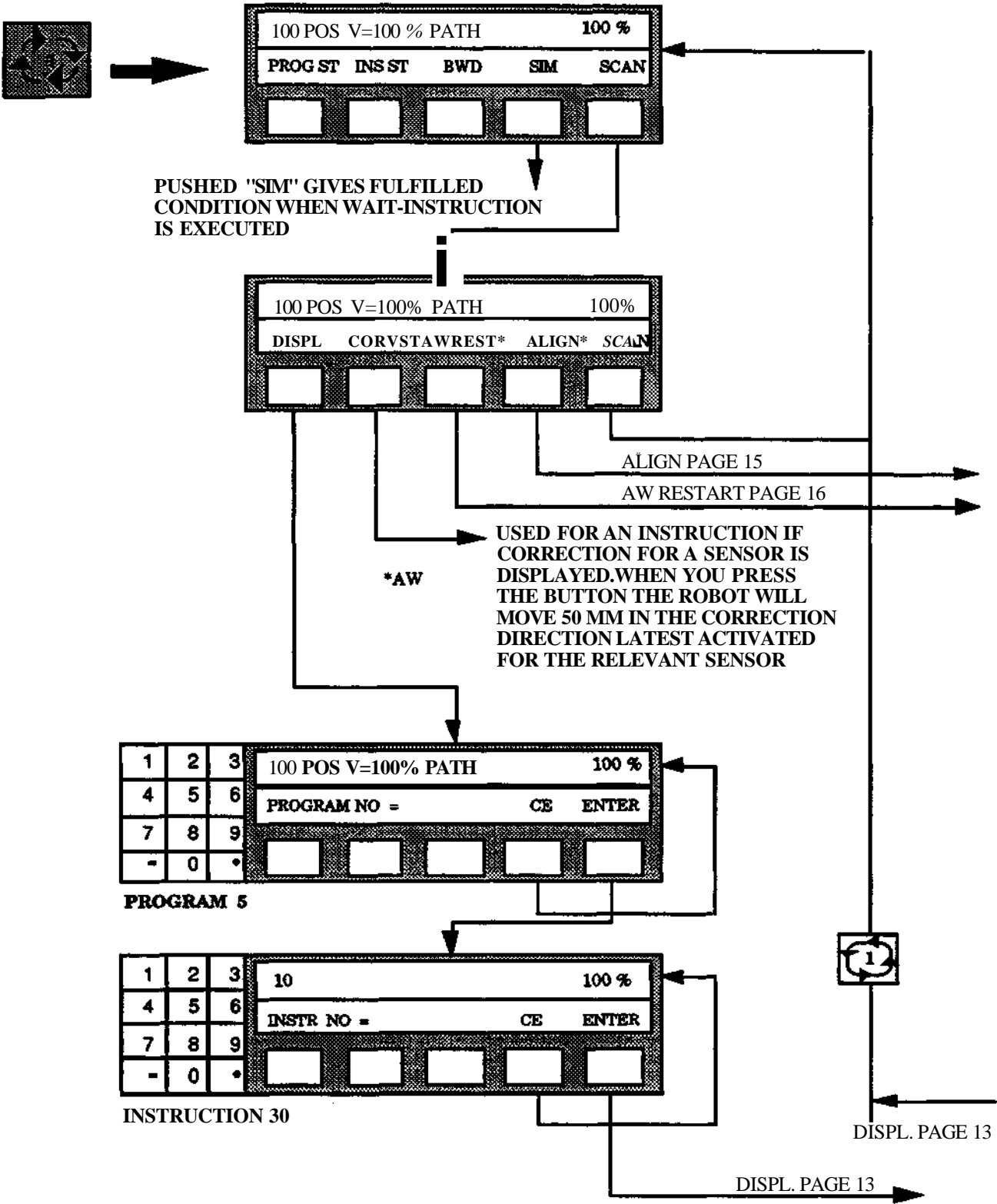


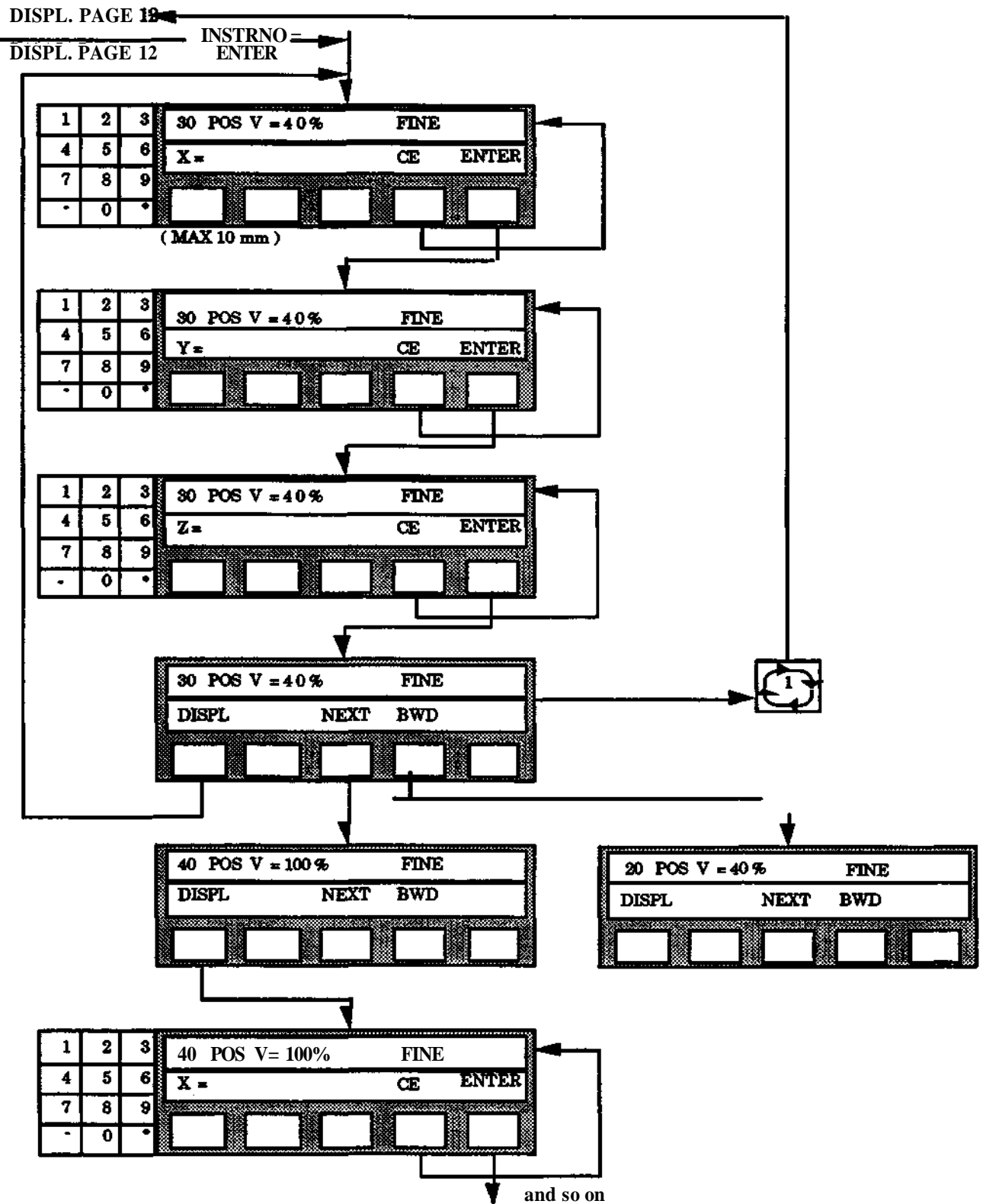
7 Automatic menu

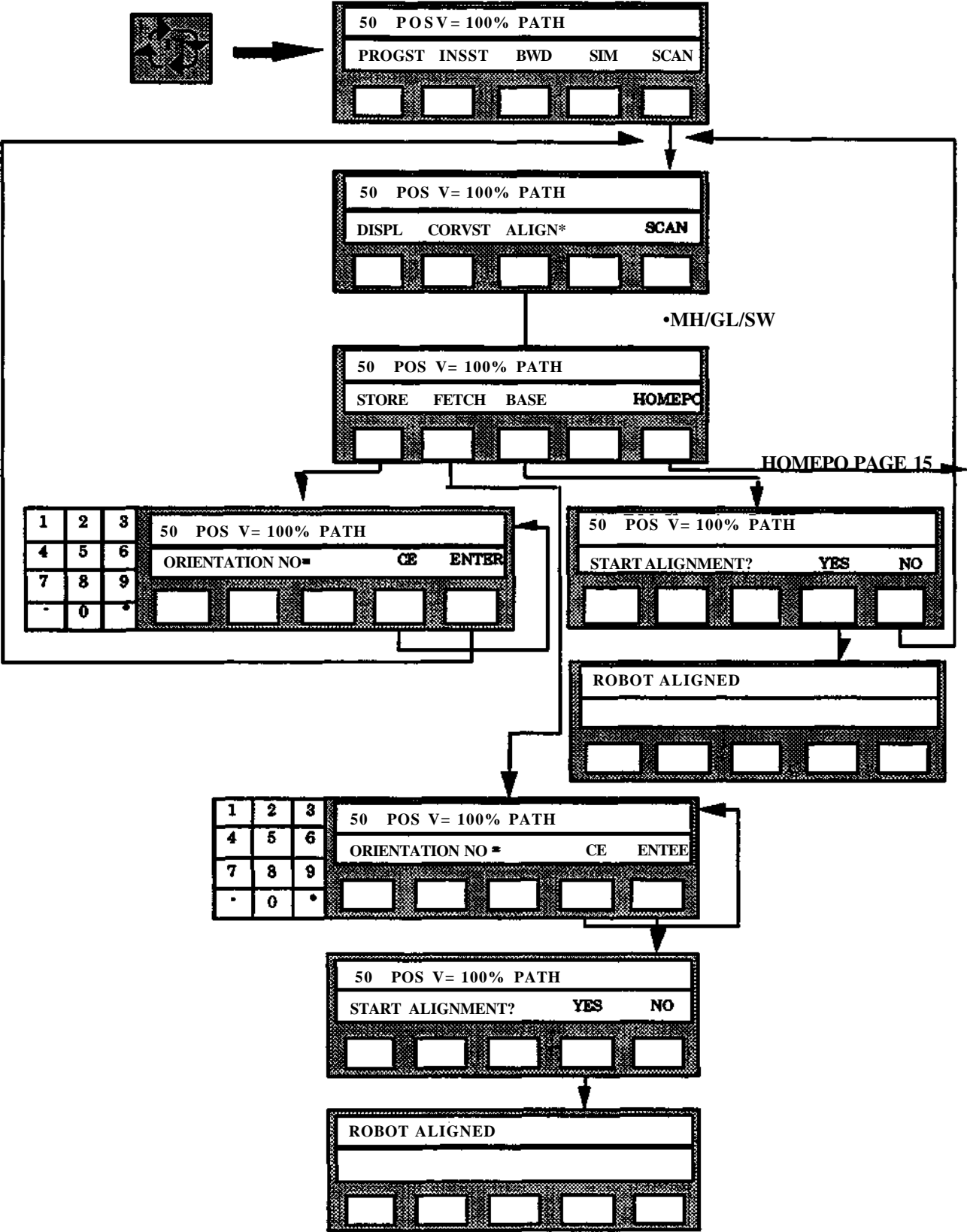


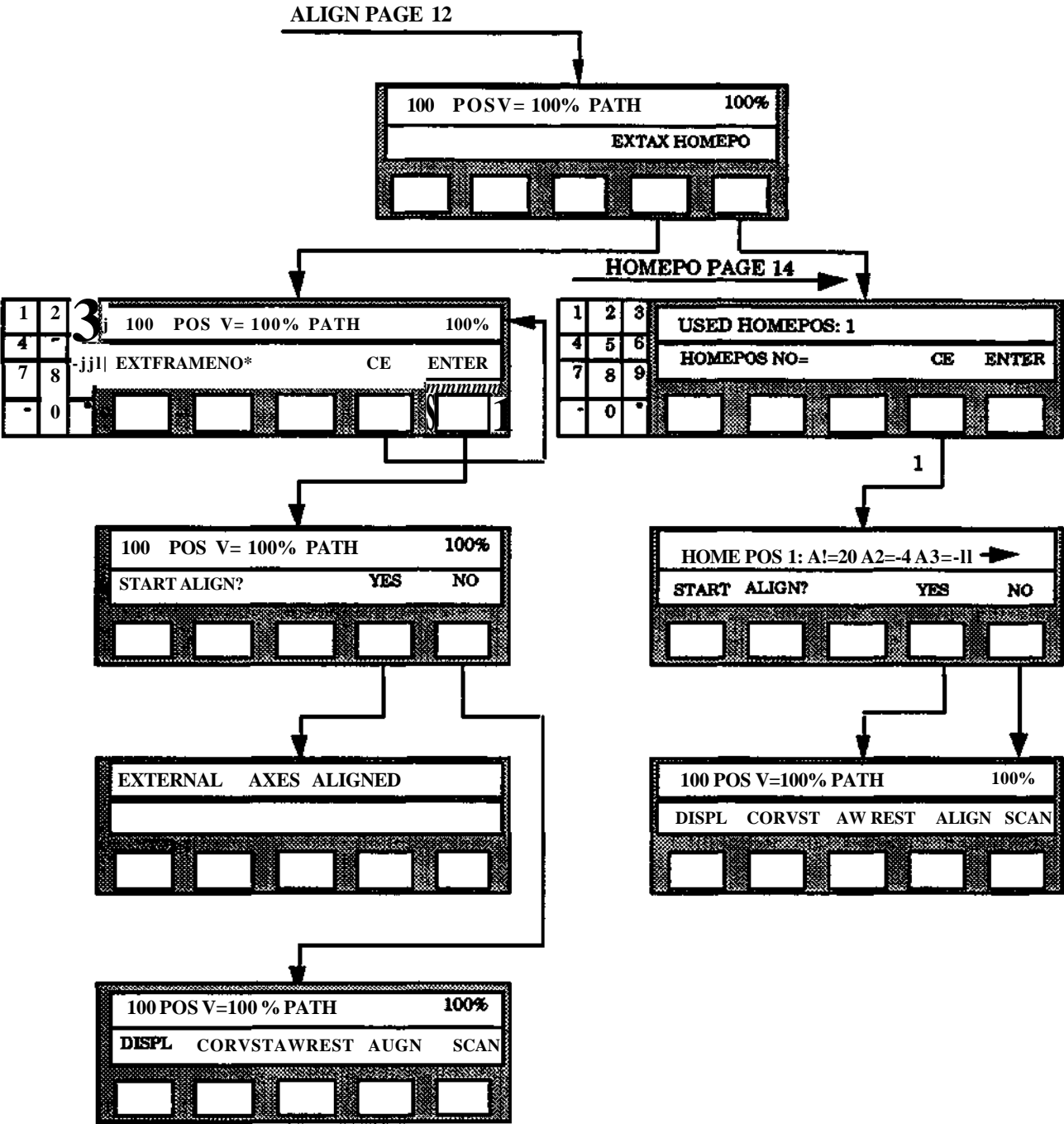
START P. PAGE 10

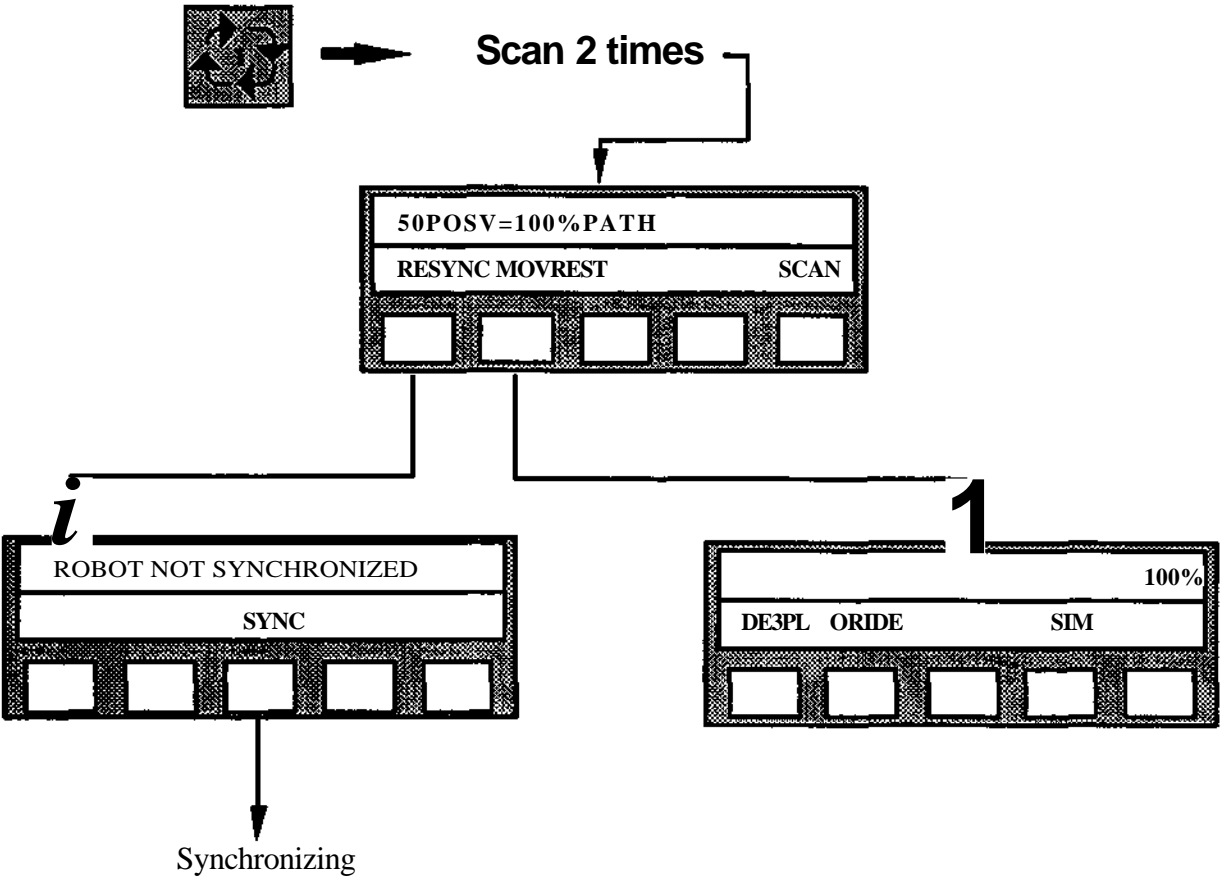








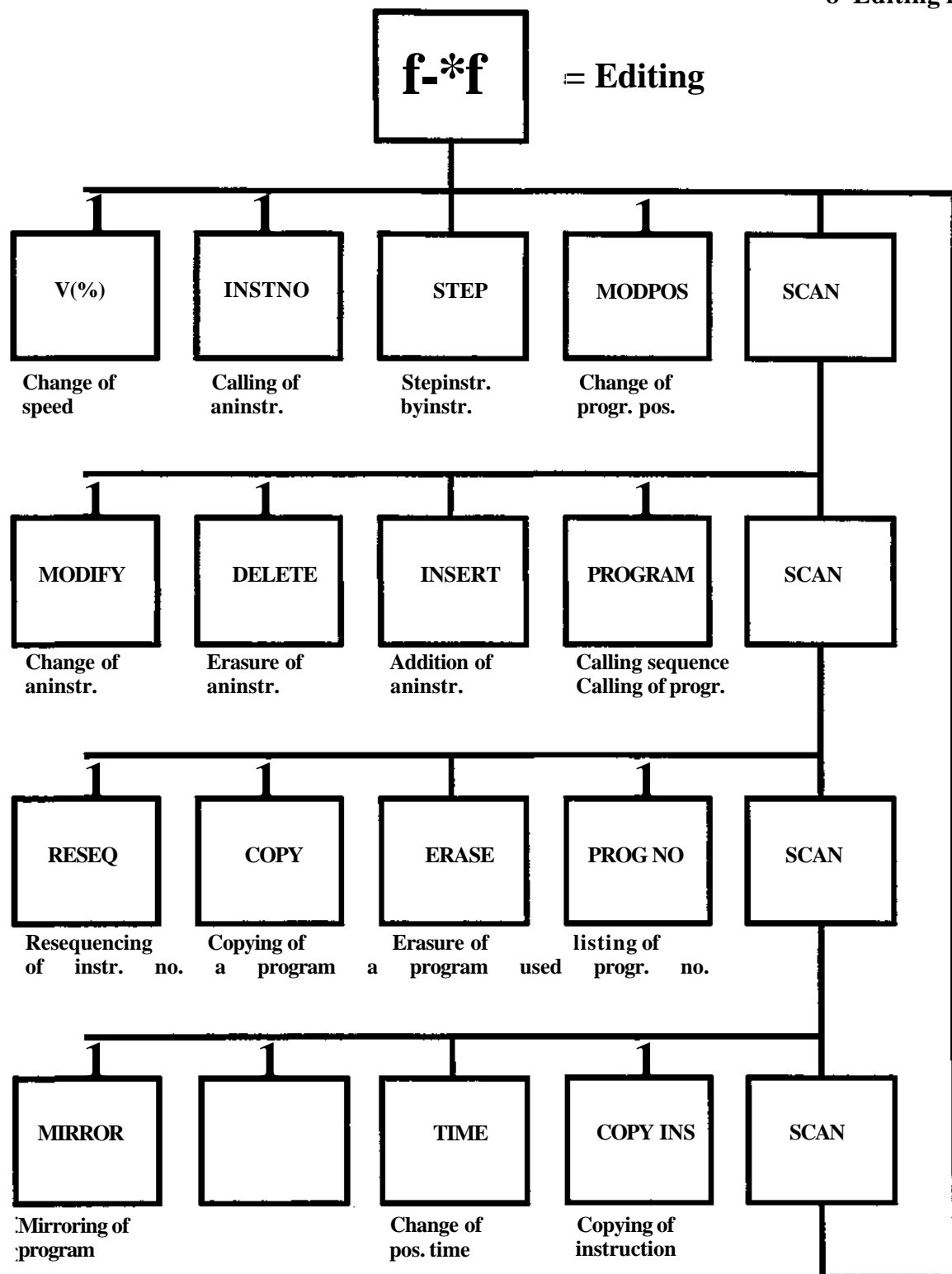




8 Editing menu

Section	Page
8.1 V%	8:5
8.2 INSTNO	8:5
8.3 STEP	8:5
8.4 MODPOS	8:5
8.5 MODIFY	8:6
8.5.1 HANDCHK	
8.5.2 MODIF and MODARG	
8.5.3 DISPL	
8.5.4 MODEXT	
8.6 DELETE	8:8
8.7 INSERT	8:8
8.8 PROGRAM	8:9
8.9 RESEQ	8:9
8.10 COPY	8:10
8.11 ERASE	8:11
8.12 PROG NO	8:11
8.13 MIRROR	8:13
8.14 TIME	8:14
8.15 COPY INS	8:15







8.1 V (%)

Change of speed

Menu: EDITING

Function: V%

Means:

The percentage speed in an instruction is changed. For changing of the basic or maximum speed, see 8.5.

Used:

When editing a program.

Executed:

After the conclusion of the procedure, the robot reacts immediately.

Procedure:

Change of percentage speed

1. Ensure that the instruction required is presented on the display.
2. Select V% under the EDITING menu.
3. Specify the new percentage speed 0 - 799.9% as a percentage with one decimal.

The new percentage speed is shown in the instruction.

8.2 INSTNO

Instead of stepping forward to a certain instruction, it is possible to call it up directly by pressing INST NO. The robot then asks which instruction number is required and when this is given, the instruction line is shown on the display.

8.3 STEP

The STEP function is used when, for example, the program is to be presented instruction by instruction. When STEP is pressed, it is possible to press INST NO FORW or BWD as required. For each pressing of these buttons, a new instruction line is presented on the display.

8.4 MODPOS

Procedures:

Redefinition of location by running the tool

Menu: EDITING

Function: MODPOS

1. Ensure that:
 - The tool moves as programmed to the position to be changed.
 - The correct instruction is shown on the display after the positioning.
2. Run the tool to the correct position.
3. Select MODPOS under the EDITING menu.
4. If the parameter MODPOS is defined (see Installation S3). Answer YES to verify the change.

If the axes 10 - 12 are present in the system "MODPOS" will affect both instructions, if "MODPOS" is selected when the POS-instructions are displayed on the programming unit. The procedure is now completed.

8.5 MODIFY

Change of instruction

Menu: EDITING

Function: MODIFY

Means:

An instruction is changed, partly or completely.

Facts:

Different subfunctions are available:

- HANDCHK
- MODINST, the complete instruction is reprogrammed.
- MODARG, a motion instruction argument except position and speed is reprogrammed.
- DISPL, the position for a motion instruction is changed.
- MODEXT, modification of external axes.

With AW software a subfunction CLE LO (clear local override) is also available. See Override, section 12.7.2.

Note **Change only absolute positions in the program, i.e. deactivated reference point and reference frame 0 active. Otherwise, the expected change of position does not correspond to the actual.**

8.5.1 HANDCHK

There is an implicit argument to the POS-instruction, HANDPOS, which stores information of the configuration of the robot arm as it was when the POS-instruction was programmed. This information is used when running the robot in Robot Coordinates or Modirect Coordinates to ensure a specific configuration (see chapter 3.4) of the arm. When running in rectangular coordinates the robot will choose the configuration corresponding to the smallest movement from start to end position. In rectangular coordinates a supervision is available, which will stop the robot if anything of the following occurs:

1. The fourth axis is more than 45 degrees in direction from its position at programming time.
2. The sixth axis is more than 90 degrees from its position at programming time.
3. The programmed axis configuration is not achieved.

This supervision is normally active, but can be deactivated and then reactivated with the function HANDCHK

If the inspection is not to be performed, "*" will be displayed in the instruction.

Change of HANDCHK argument

1. Call up the positioning instruction which is to be changed.
2. Select CHANGE under the EDITING menu.
3. Select HANDCHK
4. Answer YES or NO to the question WRIST CONFIGURATION CHECK?

8.5.2 MODINST and MODARG

Procedure:**Change of instruction**

1. Call up the instruction to be changed.
2. Select MODIFY under EDITING menu.
3. Select either MODINST or MODARG.
4. Reprogram the instruction in the usual way.

The procedure is now complete.

8.5.3 DISPL

Definition of position by entry without moving the tool

1. Call up the correct program and instruction on the display.
2. Select MODIFY under the EDITING menu.
3. Select DISPL.
4. Specify the displacement from the present location in mm max 10 mm with one decimal for X-, Y- and Z-coordinates in the base coordinate system. If any of the displacements are to be 0, press ENTER directly when the coordinate value is requested by the system.

When the menu, shown before point 3 above is performed, is presented again, the procedure is complete.

8.5.4 MODEXT

Modification of external axes

Means:

MODEXT is a function with which it is possible to copy the position of an external axis to another, within one and the same POS-instruction. It is possible to execute the copying within a sequence of instructions.

The system copies the position from the required external axis to another external axis immediately after the procedure is concluded. The possibility of displacement is available.

Procedure:

Copying of external axes

1. Call the program and the first instruction (of several) which is to be modified.
2. Select MODEXT under the EDIT/MODIFY menu.
3. Specify the axis from which the position is to be copied.
4. Specify the axis to which the position is to be copied.
5. State the number of instructions up to and including the last to be modified.
6. State the displacement in increment. (Max = working range).

When the modification is completed the instruction most recently changed is the current instruction.

Any intermediate logical instructions and POS-instructions which do not include the external axes concerned are not affected.

8.6 DELETE

Erasure of an instruction

Menu: EDITING

Function: DELETE

Means:

An instruction, or several, in a sequence in the program is erased.

Executed:

The system reacts immediately after the procedure is concluded.

Procedure:

Erase of instruction

1. Call the program required and the first instruction (of several) which is to be erased.
2. Select DELETE under the EDITING menu.
3. Specify the number of instructions which is to be erased, from and including the instruction displayed.

The first remaining instruction in the program (or the message PROGRAM END) is presented, thereby concluding the procedure.

Note.

In order to regain memory space you must clear the program block, see Clearing of program block, (see section 9.1) or make a program selection.

8.7 INSERT

Addition of an instruction

Menu: EDITING

Function: INSERT

Means:

Addition of instructions at an optional place in an existing program.

Facts:

When instructions are added, the new instructions will be numbered in accordance with the following system.

- If the next multiple often is vacant, the instruction will receive this number.
 - If the next multiple often is occupied, the next following vacant number will be used.
- See example next page:

Existing program	Instructions to be inserted	Program after insertion
:	^ ^ f o s	110
110	POS	120 POS
140	POS	130 POS
:	POS	131 POS
:		

Procedure:

Addition of instruction

1. Call the instruction after which the addition is to be made, in the required program.
2. Select INSERT under the EDITING menu.
3. Specify the number of the new instruction.

Programming in the normal way of the instructions which are to be added can now begin.

8.8 PROGRAM

To write a robot program, begin by selecting a program number by pressing PROGRAM in the editing menu after which the robot system asks for the required program number. When the number is specified, line 10 without an instruction is shown if a new program is created. If the program already contains instructions, the first program line is presented.

8.9 RESEQ

Numbering in even tenths of instructions

Menu: EDITING

Function: RESEQ

Means:

Renumbering of the instructions within a corrected program so that all instructions again have numbering in consecutive multiples of 10.

Used:

After editing of a program

Executed:

The system reacts immediately after the conclusion of the procedure.

Procedure:

Numbering of instructions in multiples of 10

1. Call the program required.
2. Select RESEQ under the EDITING menu.

The procedure is now concluded.

Otherwise:

When the function RESEQ is executed, the jump addresses within the program are also changed.

An error message is presented when a jump is to be made to an erased instruction.

The renumbering function RESEQ is associated with a function parameter with the same name. Renumbering is not permitted when the parameter is set to the value 0.

8.10 COPY

Copying of a program

Menu: EDITING

Function: COPY

Means:

A program existing in the user memory is copied under another number.

Used:

For example, when two or several similar programs are to be created.

Procedure:

Copying of a program

1. Call the program to be copied.
2. Select COPY under the EDITING menu (used program number is shown with the active program marked-up with "*" beside the number. Empty memory space, in percent of the total memory area, is also displayed.).
3. Specify the program number 0- 9999 under which the copy of the program is to be stored in the user memory. (Attempts to copy a program under an already occupied program will cause an error message.)

The procedure is now completed.

8.11 ERASE

Erasure of a program

Menu: EDITING

Function: ERASE

Means:

One or several programs are to be erased from the user memory.

Facts:

When the function is selected, the size of the remaining user memory can be presented as a readout by pressing the SHIFT button.

Executed:

The system reacts immediately after the procedure is concluded by specifying which program is to be erased.

Procedure:

Erasure of a program

1. Select ERASE under the EDITING menu (the numbers of program used are presented with the active program marked with an asterisk. Empty memory space, in percent of the total memory area, is also displayed.).
2. Give the program (or the first program in a continuous number series) to be erased with the number 0 - 9999.

If only this program is to be erased, press ENTER. Thereby concluding the procedure. Otherwise continue with point 3 below.

3. Specify the last program number, 0 - 9999, in the number series of the program to be erased.

The procedure is now concluded and the system erases the programs specified, regains the space after the erased programs in the memory and responds with information about which programs that have been erased.

8.12 PROG NO

All numbers of programs stored in memory can be displayed by pressing PROG NO. All program numbers in use are then presented on the display. An active program is marked with an asterisk after the program number. Empty memory space is also displayed.

8.13 MIRROR

Mirroring of program

Menu: EDITING

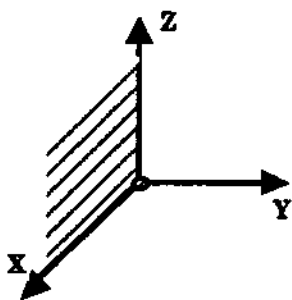
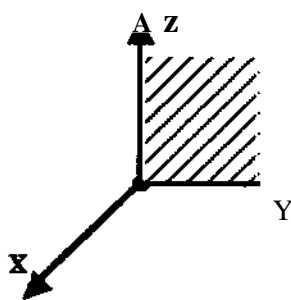
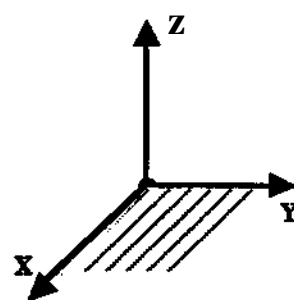
Function: MIRROR

Means:

All the positions of the active program are reflected in relation to a mirroring plane.

Facts:

The program is reflected in base coordinates. Three different planes are available:

XZ-plane**YZ-plane****XY-plane**

In addition, a displacement in base coordinates can be added to each of the planes.

Note.

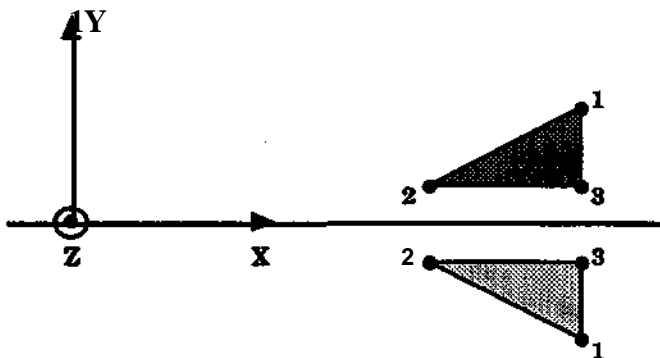
The robot tool must be designed so that it can also be used, in the same way as before, in a mirrored program i.e. it should be symmetric around the selected plane.

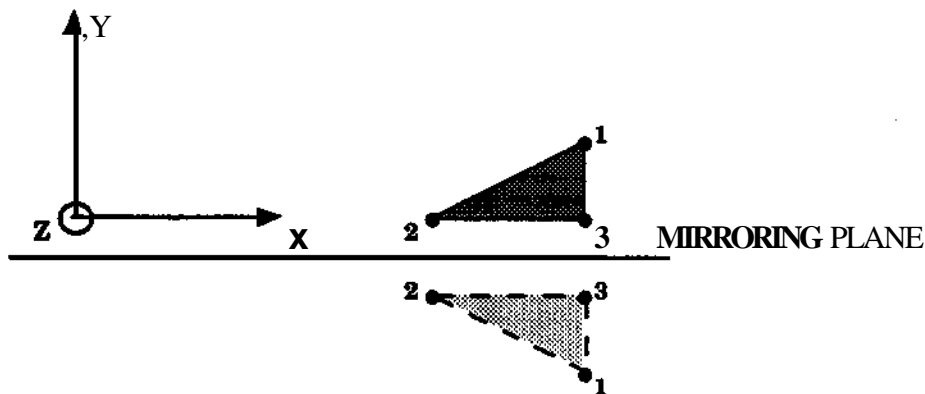
Used:

When a (copied) program is to be mirrored (in relation to the original program).

Executed:

A mirroring of a program is performed immediately after the conclusion of the procedure. Two examples of mirroring are shown below:

Example 1: Mirroring in XZ-plane

Example 2: Mirroring in parallel displaced XZ-plane**Procedure:**

1. Select the program to be mirrored.
2. Select MIRROR under the EDITING menu.
3. CLEAR CONFIG YES NO. (from M93/5, otherwise continue with 4)

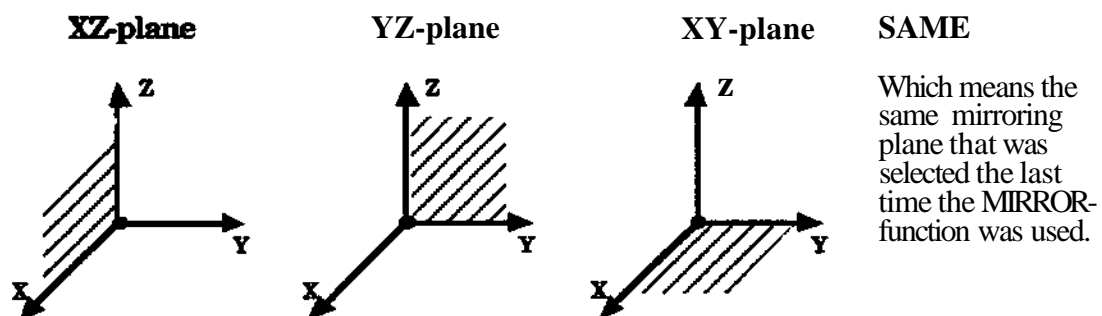
Answer Yes and the robot will take the shortest way to get the right orientation of the tool. This means that axis 1,4 and 6 can be rotated 180 degrees compared to the original program (performance as in control programs up to and including M93/4).

Answer No, and it will try to have axes 1,4 and 6 in approximately the same position as they had when the program was not mirrored. Depending on the orientation it is not always possible to not clear the configuration.

When you once have answered YES, you can not restore the configuration.

Hint: Try to mirror the program by answer NO. If you later on want to clear the configuration, mirror the program twice and clear the configuration when you mirror the program.

4. Select mirroring plane according to the figures below:
5. Press YES if a parallel offset is wanted.



6. Press NO if no parallel displacement is wanted.

In this case the mirror plane is defined by the base coordinate system like in example 1 above. The mirroring will be executed immediately.

A displacement can be defined in two ways, A, by entering a value from the teachpendant,

8 Editing menu

or B, by moving the robot a corresponding distance.

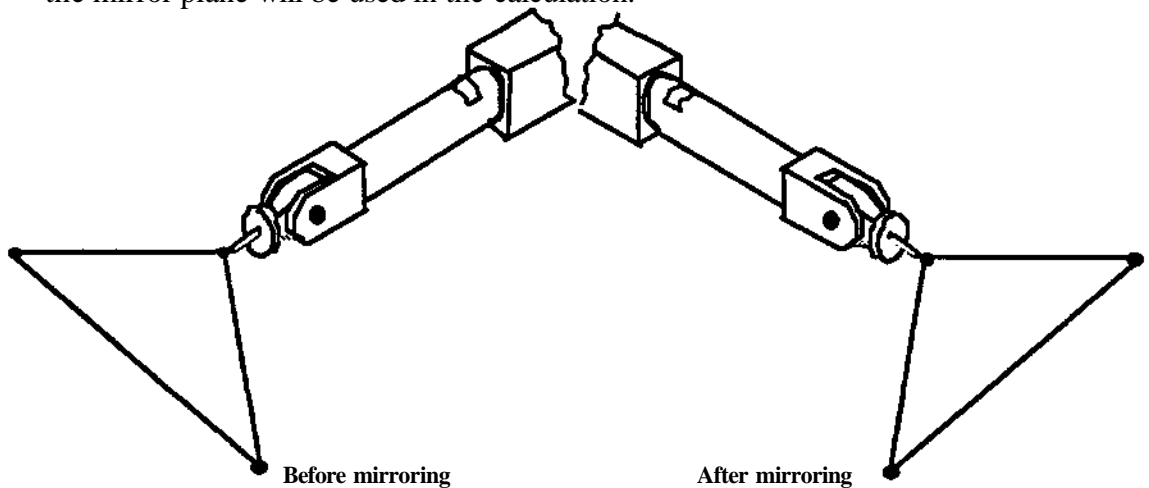
- A** Enter the displacement concerned along the relevant axis in the basic coordinate system. Use the correct sign for the value which is to be expressed in mm.

In example 2 above a negative value should be entered, corresponding to the distance from origo to the mirroring plane in negative y-direction.

- B** Position the robot to a position in the original program with the joystick and press ST POS. Run then to the corresponding position in the "mirror image" of the program and press ENDPOS.

In example 2 above the robot is first positioned, for instance, in the point 3 in the original program and STPOS is pressed. After that the robot is manually moved to the point 3 where it should be located in the reflected program and ENDPOS is pressed. Then the mirrorplane will be automatically calculated and all positions in the program will be mirrored.

Note, when placing the robot in the mirrored point, only the distance perpendicular to the mirror plane will be used in the calculation.



8.14 TIME

Change of positioning time

Menu: EDITING

Function: TIME

Means:

The positioning time in an instruction is changed.

Used:

When editing a program.

Executed:

After the conclusion of the procedure, the system reacts immediately.

Procedure:**Change of positioning time**

1. Ensure that the instruction required is presented on the display.
2. Select TIME under the EDITING menu.
3. Specify the new positioning time in seconds with two decimals (0.00 - 650.00 s).

The new positioning time is shown in the instruction.

8.15 COPY INS**Copying of instruction****Means:**

One instruction, or more, are copied.

Facts:

The copied instructions are placed after the current instruction.

Copying to a place between two instructions can be performed if enough spare instruction numbers are available.

Instructions can be copied from another program.

Performed:

The system reacts immediately after concluded procedure.

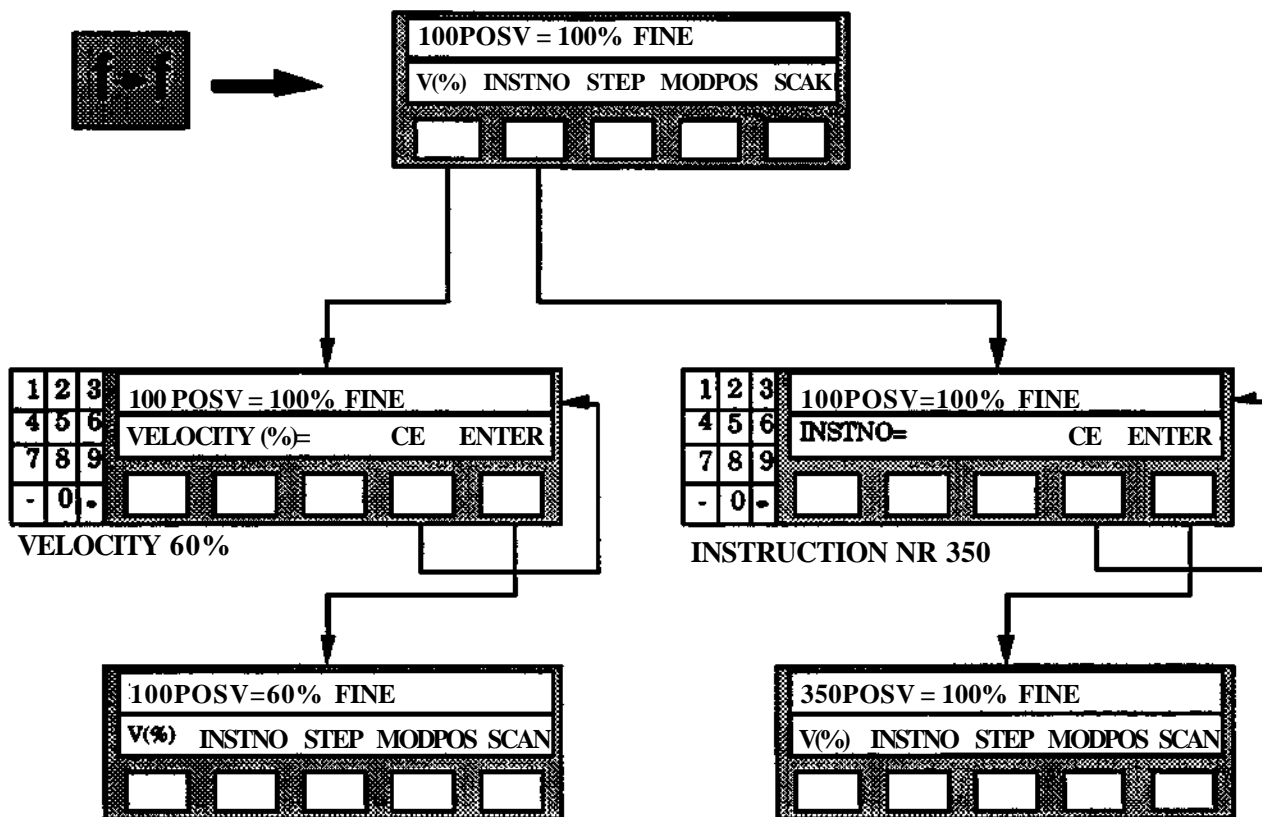
Procedures:

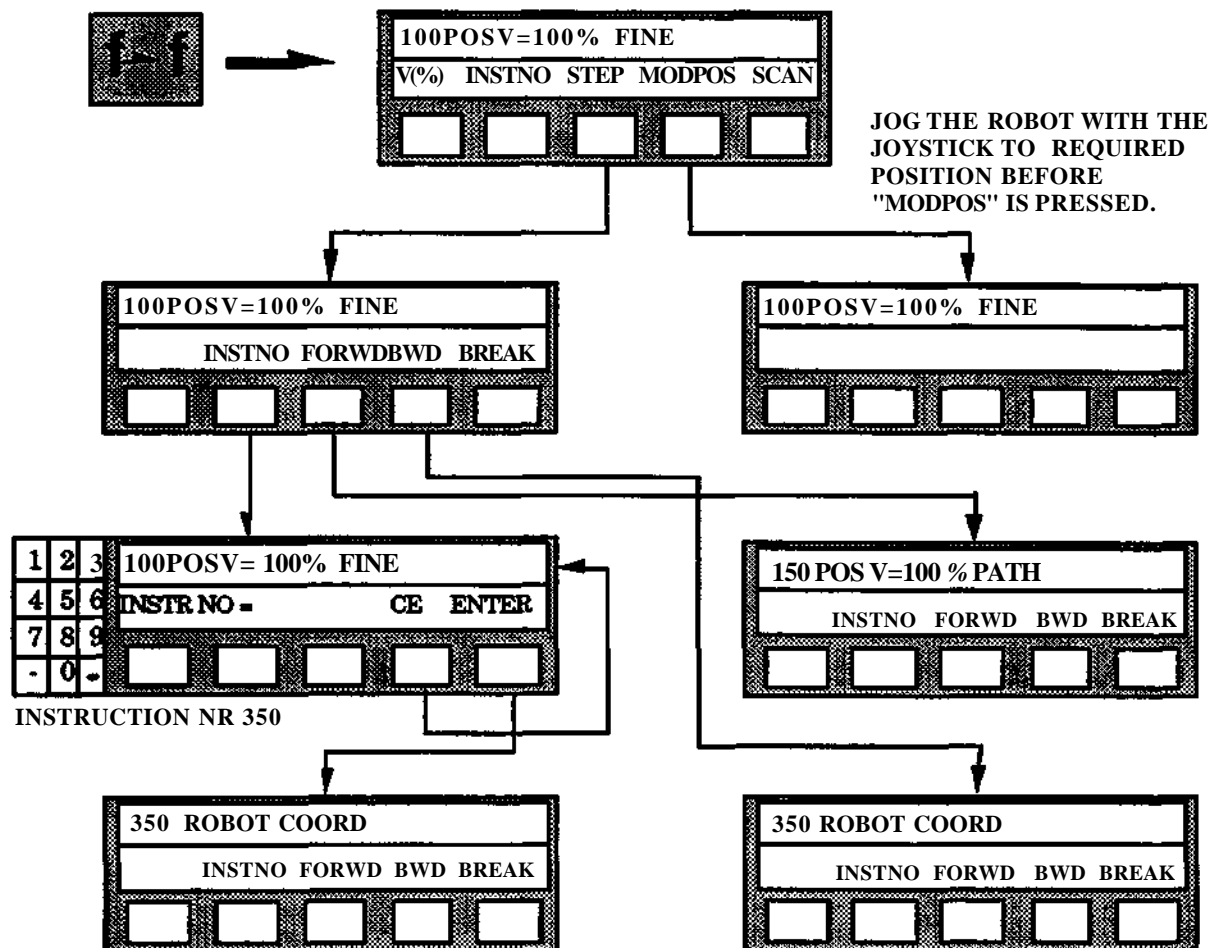
Menu: EDITING

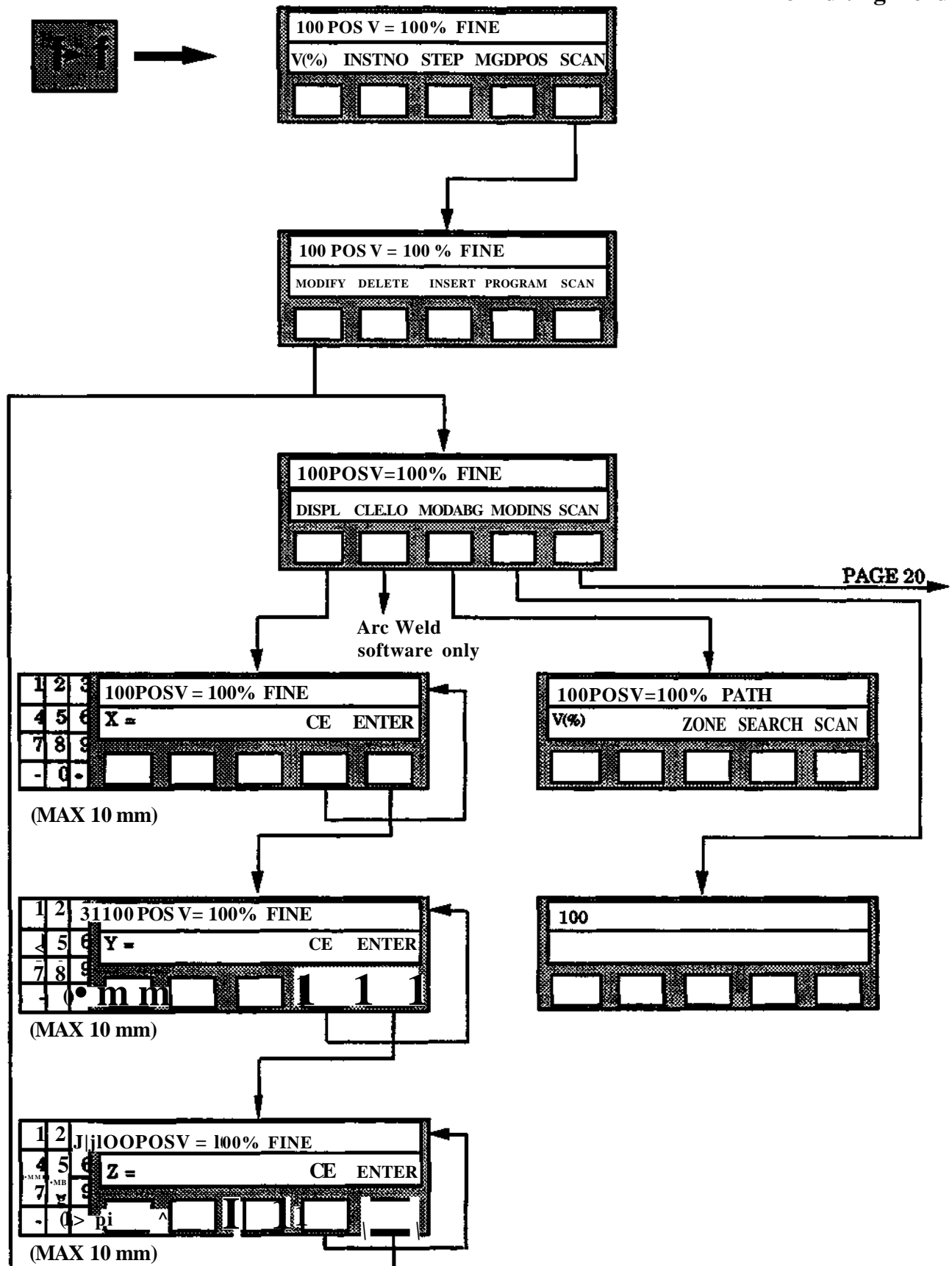
Function: COPYINS

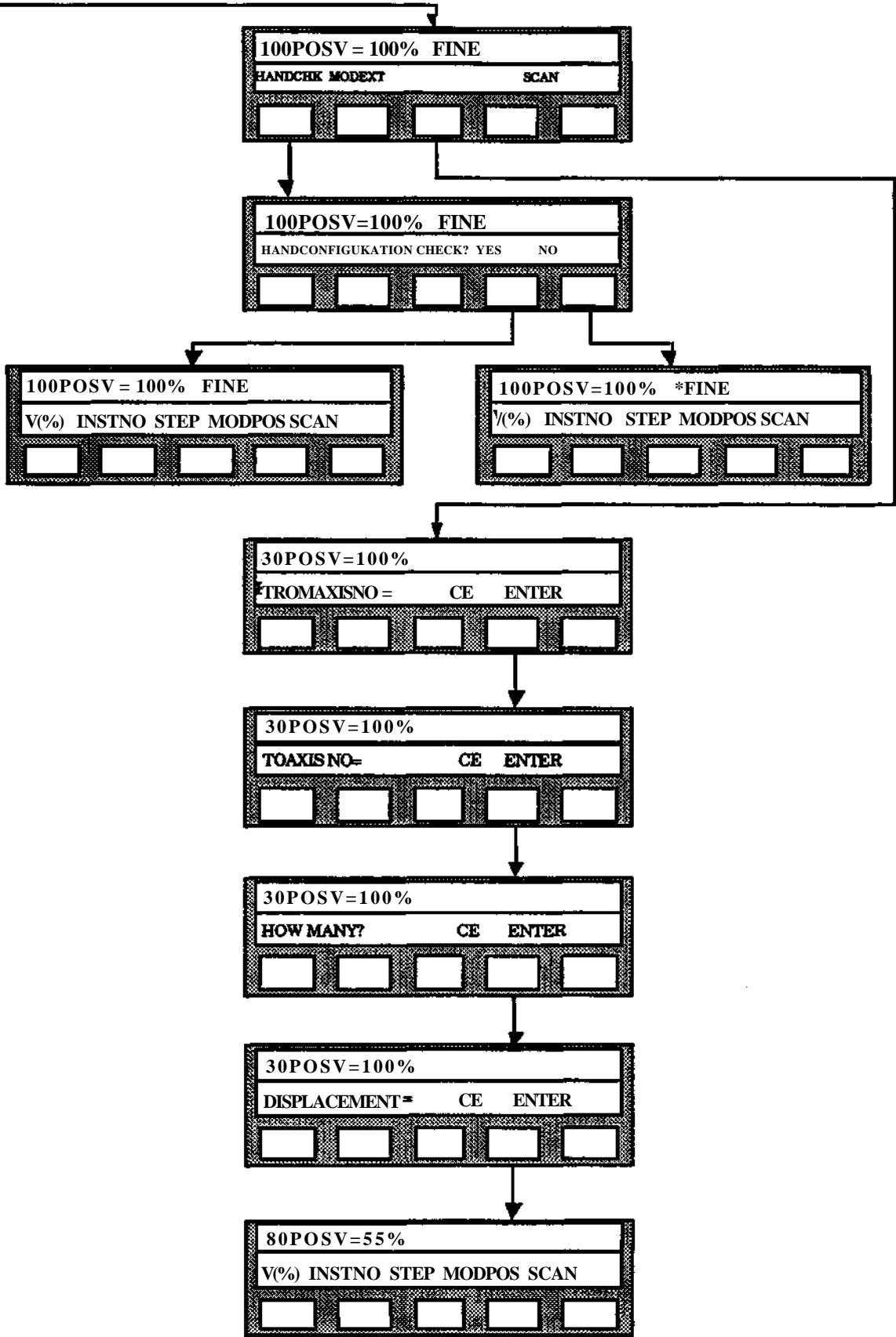
1. Call the instruction, after which the copied instructions are to be placed.
2. Select COPYINS under the EDITING menu.
3. Specify from which program the instructions are to be copied. If they are to be copied from the current program, press ENTER directly.
4. Specify the instruction number for the first instruction to be copied.
5. Specify the number of instructions that is to be copied. If one instruction only is to be copied, press ENTER directly.

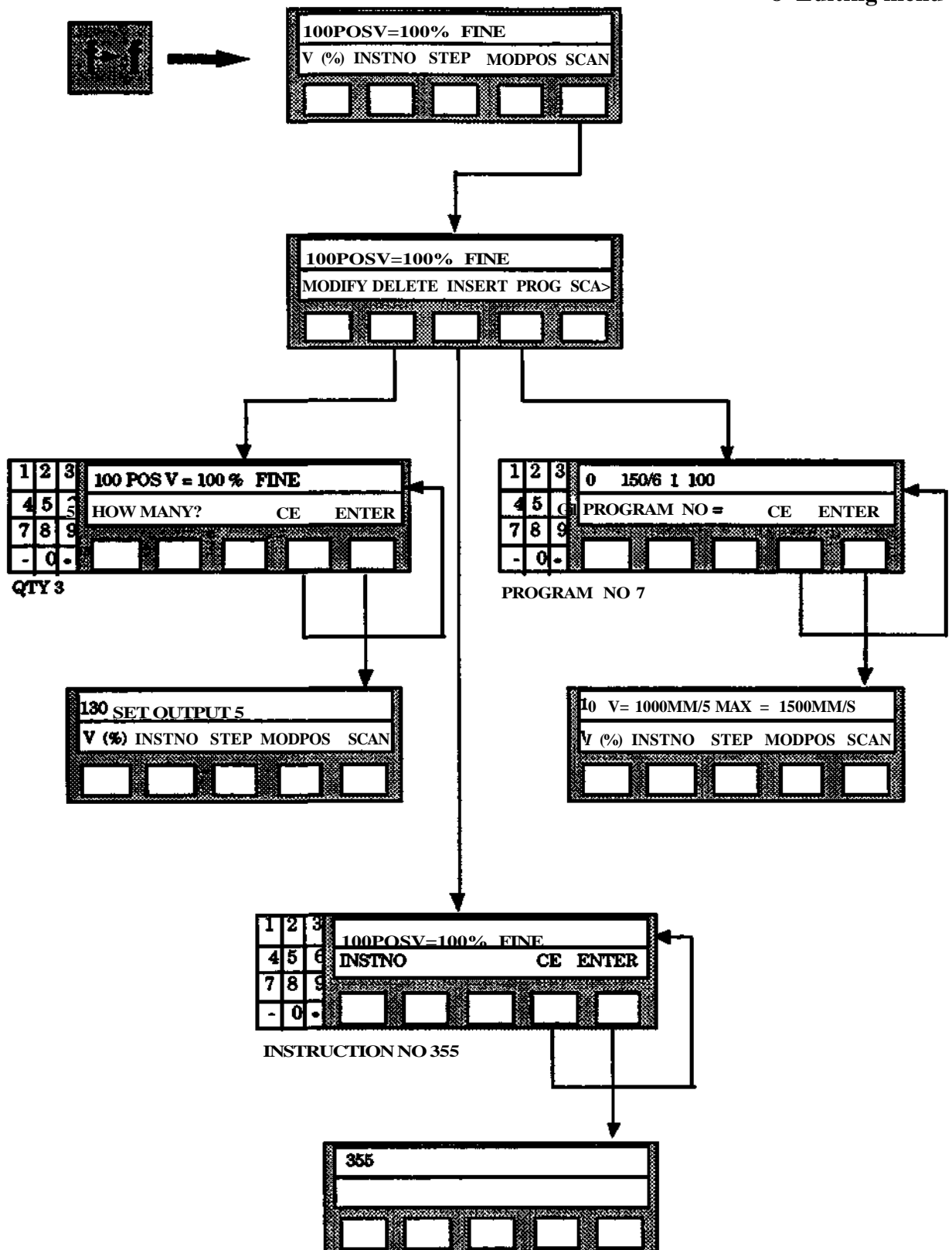
The last copied instruction is now shown and the procedure is thereby concluded.



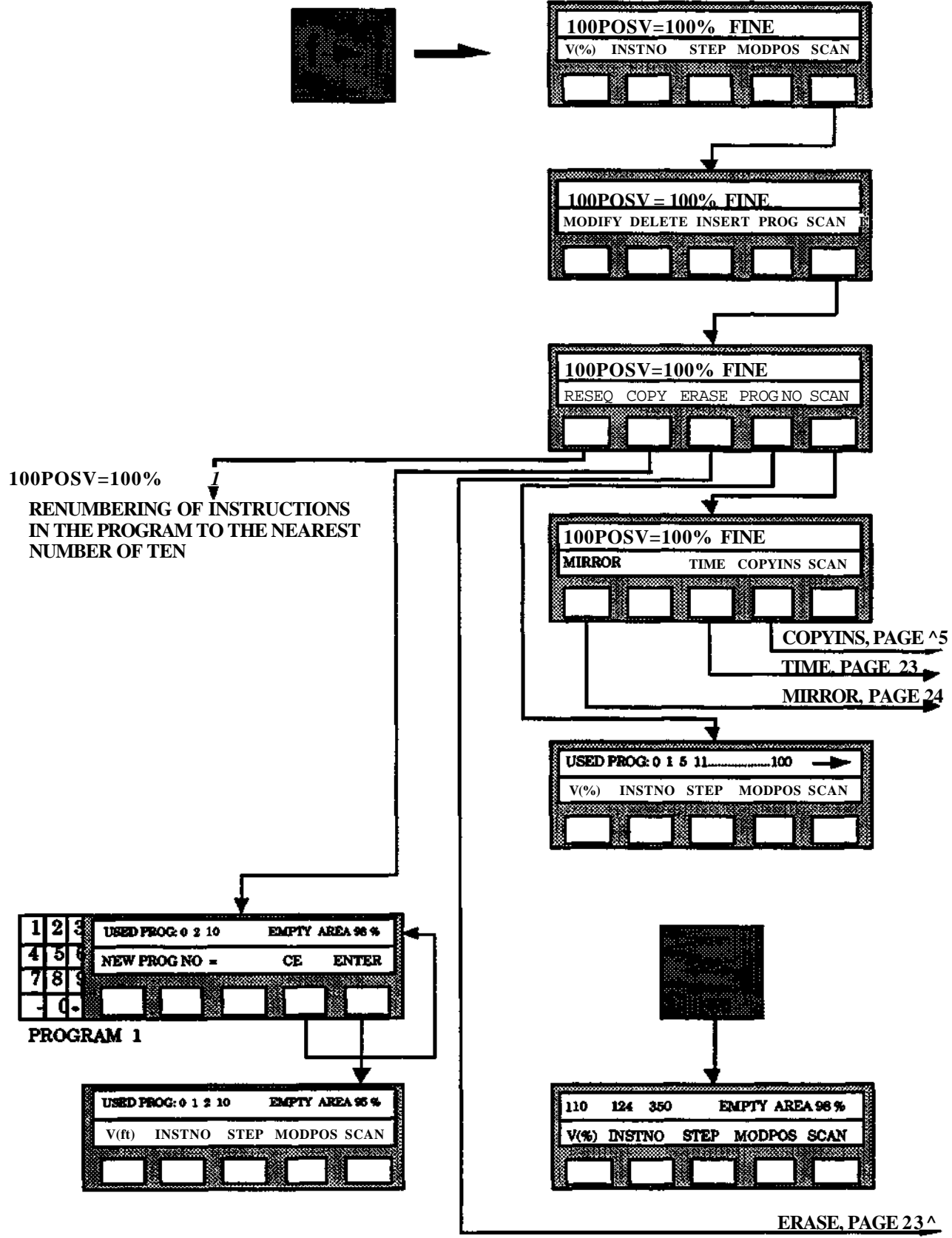




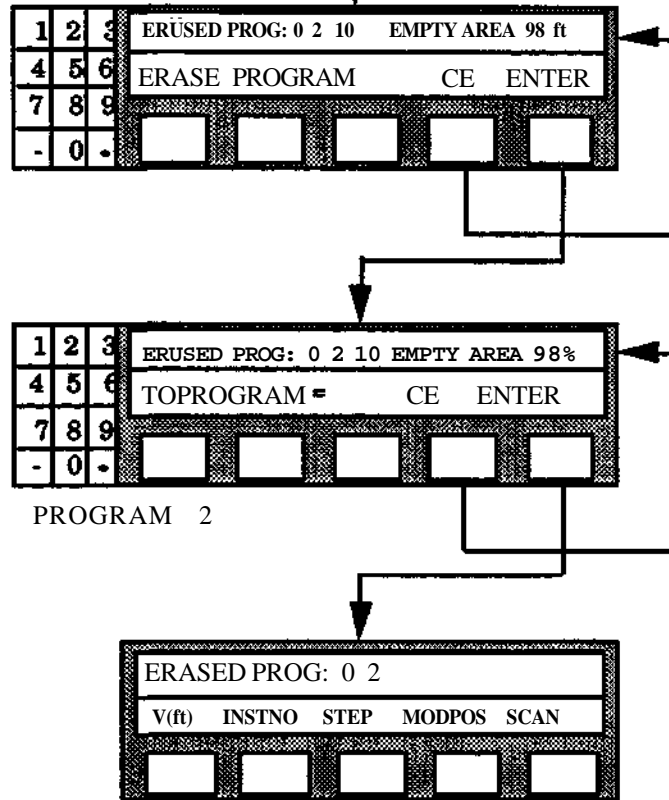




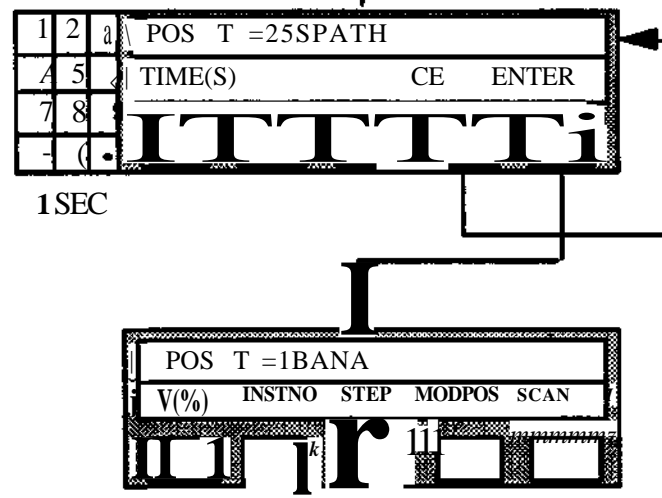
8 Editing menu

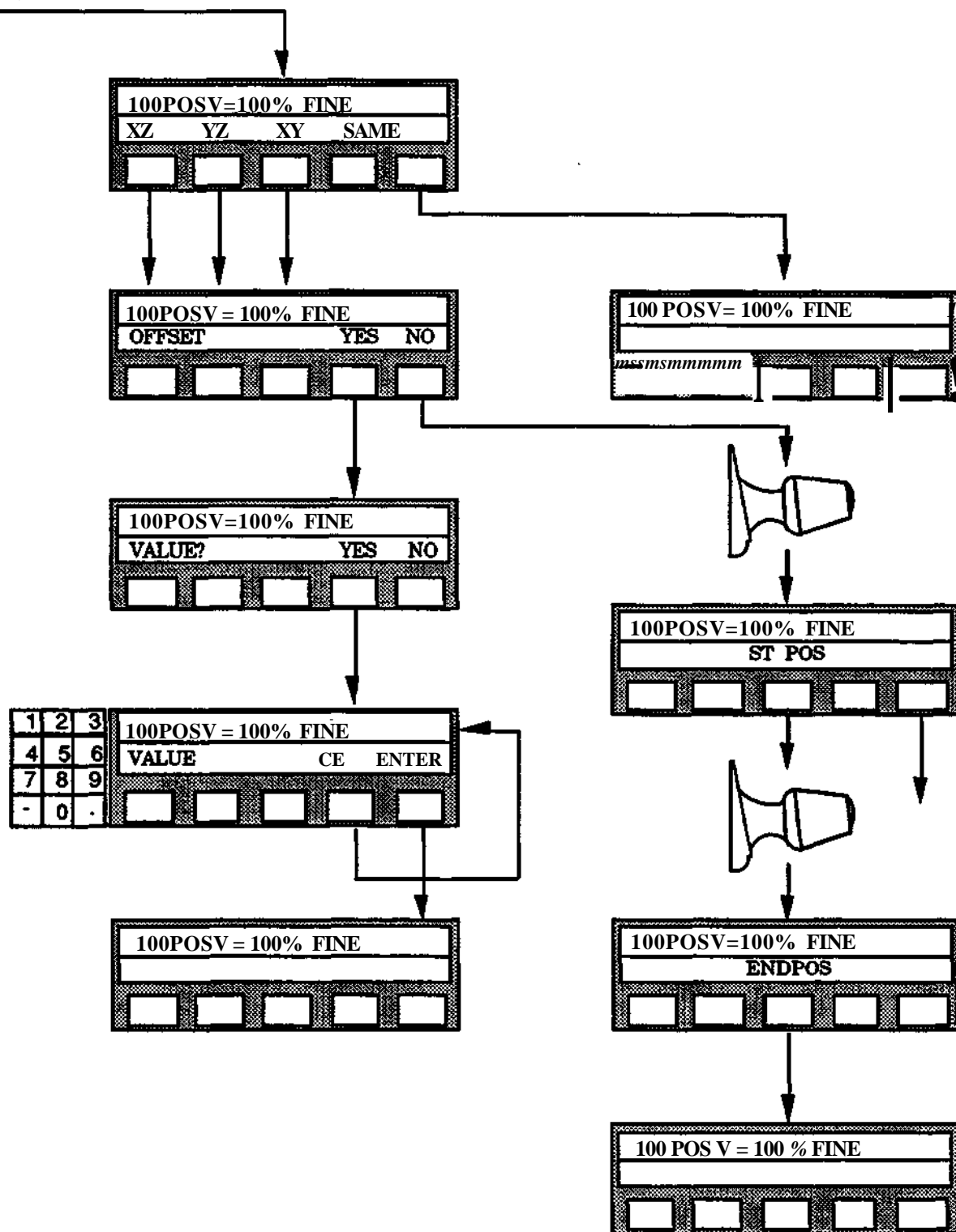


ERASE, PAGE 22

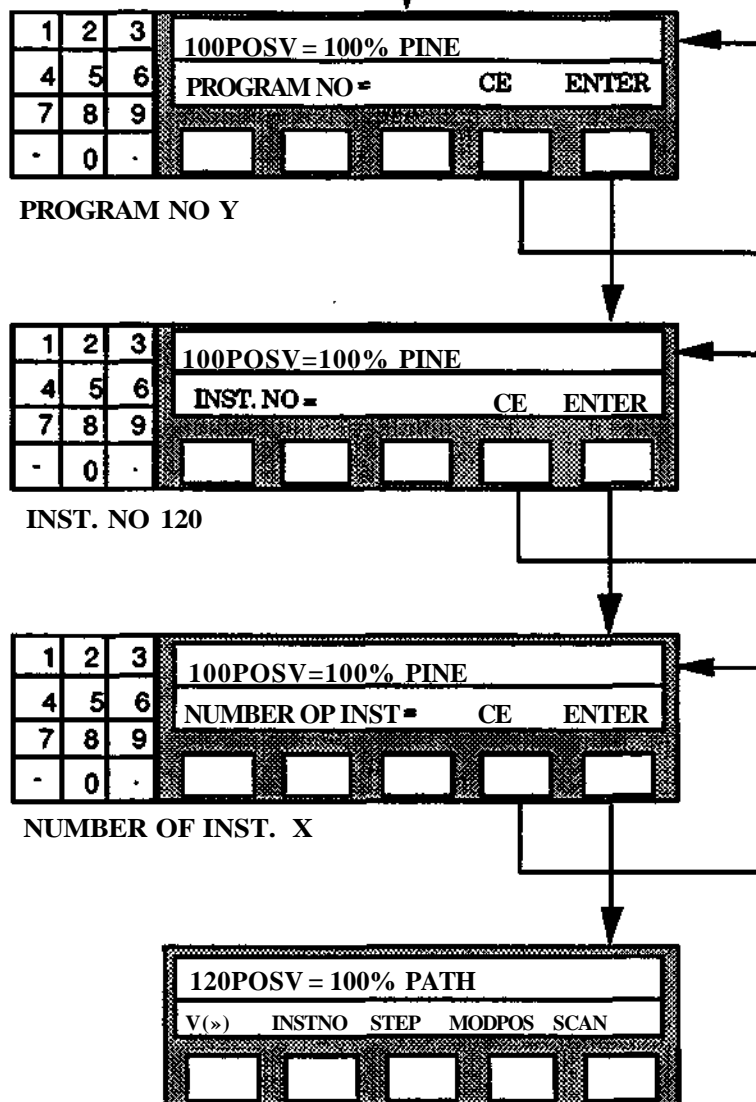


TIME, PAGE 22





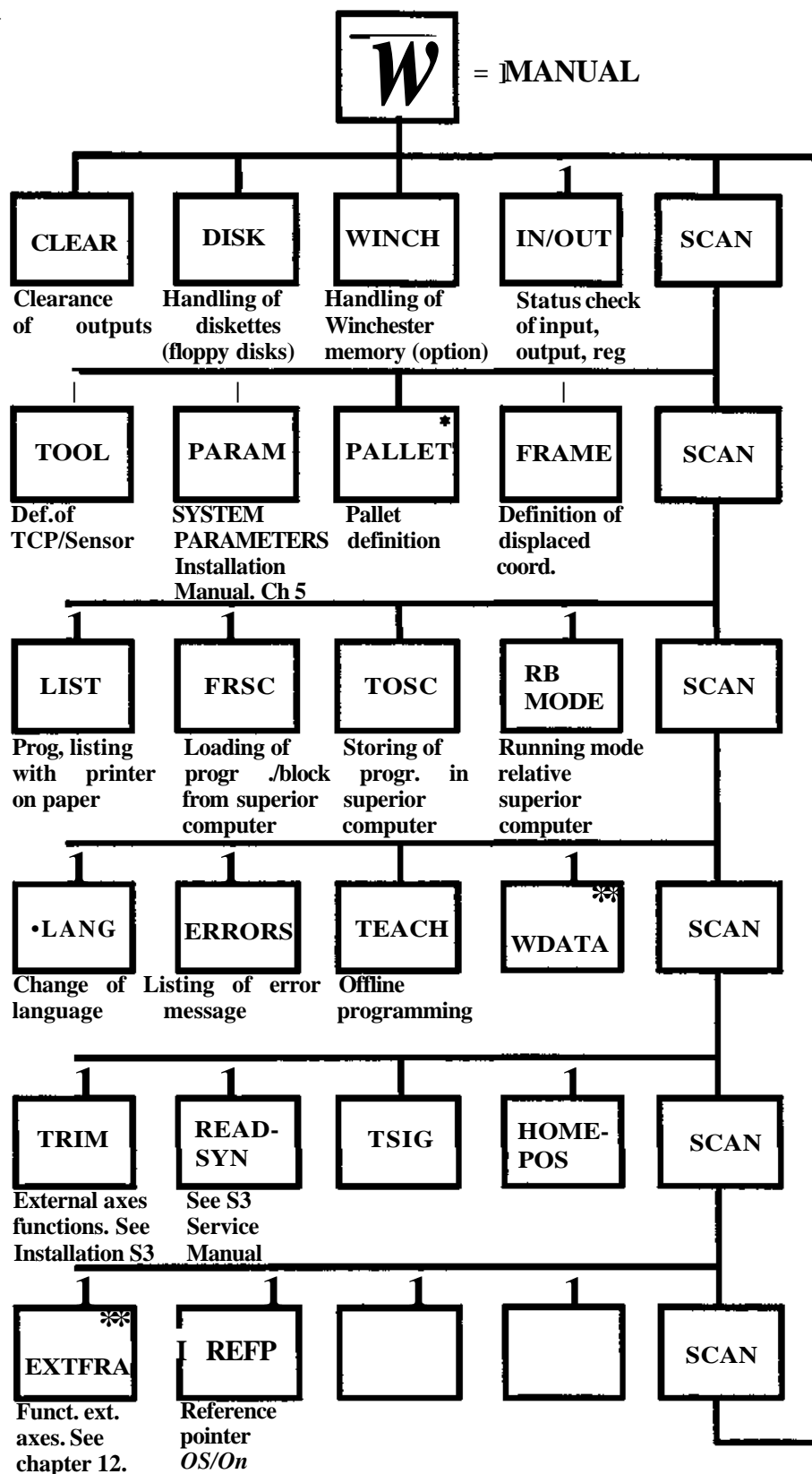
COPYINS, PAGE 22





9 Manual menu

Section	Page
9.1 CLEAR	9:5
9.2 DISK	9:6
9.2.1 DISK (FROM DISK)	
9.2.2 DISK (TO DISK)	
9.2.3 DISK (INIT)	
9.2.4 DISK (DELETE)	
9.3 WINCHESTER MEMORY	9:8
9.3.1 WINCH (FR WINCH)	
9.3.2 WINCH (TO WINCH)	
9.3.3 WINCH (INIT)	
9.3.4 WINCH (DELETE)	
9.4 IN/OUT (INPUT, OUTPUT, REG)	9:9
9.5 TOOL (TCP, SENSOR, WRIST LOAD, SOFT S)	9:10
9.6 FRAME	9:26
9.7 LIST (prog, errors)	9:28
9.7.1 PROG	
9.7.2 ERRORS	
9.7.3 SYS PAR	
9.7.4 USER PAR	
9.8 FRSC	9:28
9.9 TOSC	9:29
9.10 RBMODE	9:30
9.11 •LANG	9:30
9.12 ERRORS	9:30
9.12.1 Software Configuration	
9.13 TEACH	9:32
9.14 WDATA (arc weld function)	9:34
9.15 PALLET	9:34
9.16 EXTFRAME	9:35
9.17 HOMEPOS (from M93/5)	9:35
9.18 REFP	9:36



** AW
* MH/GL/SW

9.1 **CLEAR**

Clearing of a program block

Menu: MANUAL

Function: CLEAR

Means:

The program block is reset so that:

- The first instruction in the program 0 is presented.
- All outputs and register are cleared.
- All stored positions (with STO POS function) in position registers are cleared.
- Correction movements for all sensors are cleared, see Adaptivity.
- The system regains memory space after erased instructions.
- Program displacement by means of reference point is deactivated.
- The velocity override is set to 100%.

Used:

- When a program block is to be restarted from the beginning.
- When it is necessary to regain blocked memory space after manual erasure.

Executed:

The system reacts immediately after the procedure is completed.

Procedure:

Clearing of a program (Regaining of memory space).

Select CLEAR under the MANUAL menu, thereby concluding the procedure.

Otherwise:

The function should be used with discretion during manual work with a program after a program stop. The contents of the register and outputs can control the complete working process which has been stopped.

If a program assumes certain value at the output and register, this must be set manually before the program is started after a resetting to 0.

9.2 DISK

9.2.1 DISK (FR DISK)

The transfer of information (data) from a diskette to the robot system user memory is designated loading. A complete program block or an individual program can be loaded with the help of the programming unit.

Loading of a program block or program with the same number as one already in the user memory results in the previous program/block being written over.

Manual loading from diskette

Menu: MANUAL

Function: DISK

1. Place the required diskette in the disk drive.
2. Select DISK from the MANUAL menu.
3. Select FR DISK
4. Specify the number of the BLOCK to be loaded.
5. Select either:
 - REPLACE The complete user memory is erased and the block is entered.
 - ADD ALL The user memory contents remain intact and are supplemented with the block which is entered. The program loaded however writes over any duplicates in the user memory.
 - ADD ONLY The user memory is supplemented with one of the programs in the block specified in Point 4 after the program to be loaded has been specified. The loading is started when ENTER is pressed.
6. If ADD ALL is used, state whether arc weld data is to be loaded or not.

9.2.2 DISK (TO DISK)

A complete program block can be stored on a diskette.

Manual storing on a diskette

Menu: MANUAL

Function: DISK

1. Place the required diskette in the disk drive.
2. Select DISK from the MANUAL menu.
3. Select TO DISK
4. Specify the number of the BLOCK to be stored.

An error message is presented on the display if there is no diskette in the disk drive.

Up to 7 program blocks with different numbers can be stored on each diskette, in addition to system data. Program blocks can have numbers from 0 to 9999. Program blocks stored previously are written over if a new block with the same number is entered for storage.

The entry of the block into storage is completed when "READY" is presented on the display. When storing a block on diskette the welding data, if any, are also stored in the same block. For a description of welding data, see chapter 10.2.3. No other data are stored in the block. However TCP-registers and FRAME-registers will be stored on diskette together with systemparameters under function PARAM. See Installation Manual.

9.2.3 DISK (INIT)

A new diskette must be initialized before being used. The contents of a diskette used previously can be erased by initializing the diskette again.

Diskettes can only be initialized when the robot system is at Standby.

Initializing of a diskette

Menu: MANUAL

Function: DISK

1. Place the required diskette in the disk drive.
2. Select DISK from the MANUAL menu.
3. Select INIT.
4. The question INITIATE DISK? is presented. Start the initializing by pressing YES.

The initialization is completed when "READY" is presented.

9.2.4 DISK (DELETE)

A complete program block can be deleted from the diskette.

Manual deleting from diskette

Menu: MANUAL

Function: DISK

1. Place the required diskette in the disk drive.
2. Select DISK from the MANUAL menu.
3. Select DELETE.
4. Specify the number of the BLOCK to be deleted.
5. The question DELETE BLOCK? is presented start deletion by pressing YES.

9.3 WINCHESTER

9.3.1 WINCH (FR WIN)

The transfer of information (data) from Winchester to the robot system user memory is designated loading. A complete program block can be loaded with the help of the programming unit.

Loading of a program block or program with the same number as one already in the user memory results in the previous program/block being written over.

Manual loading from Winchester

Menu: MANUAL

Function: WINCH

1. Select WINCH from the MANUAL menu.
 2. Select FR WIN.
 3. Specify the number of the BLOCK to be loaded.
 4. Select either:
 - REPLACE The complete user memory is erased and the block is entered
 - ADD ALL The user memory contents remain intact and is supplemented with the block which is entered. The program loaded is however written over any duplicates in the user memory.
 - ADD ONLY The user memory is supplemented with one of the programs in the block specified in point 3 above after the program to be loaded has been specified.
- The loading is started when ENTER is pressed.

9.3.2 WINCH (TO WINCH)

A complete program block can be stored on the Winchester memory.

Manual storing on Winchester

Menu: MANUAL

Function: WINCH

1. Select WINCH from the MANUAL menu.
2. Select TO WIN.
3. Specify the number of the BLOCK to be stored.

Up to 250 program blocks with different numbers, 0-9999, can be stored. Program blocks previously stored is written over if a new block with the same number is entered for storage.

The entry of the block into storage is completed when "READY¹" is shown on the display.

9.3.3 WINCH (INIT)

The Winchester memory must be initialized before being used, which can only be performed when the robot system is in the STANDBY mode.

Initializing the Winchester

Menu: MANUAL

Function: WINCH

1. Select WINCH from the MANUAL menu.
2. Select INIT.
3. The question INITIATE WINCHESTER? is presented. Start the initialization by pressing YES.

The initialization is completed when "READY" is presented.

9.3.4 WINCH (DELETE)

A complete program block can be deleted from the Winchester.

Manual deleting from Winchester

Menu: MANUAL

Function: WINCH

1. Select WINCH from the MANUAL menu.
2. Select DELETE.
3. Specify the number of the BLOCK to be deleted.
4. Verify the deletion by pressing YES.

9.4 IN/OUT

IN/OUT (INPUT)

Check of the status of an input

Menu: MANUAL

Function: IN/OUT

1. Select IN/OUT under the MANUAL menu.
2. Select INPUT.
3. Give input number.

The current status of the input is read once and displayed. It is possible to:

- Read the status of the input once more with SAME.
- Present the status of the next input with NEXT.

Exit finally from this menu by selecting any other menu except POSITION.

IN/OUT (OUTPUT)**Check and possible changing of status of an output**

Menu: MANUAL

Function: IN/OUT

1. Select IN/OUT under the MANUAL menu.
2. Select OUTPUT.
3. State the output number.

The current status of the output is now displayed. It is possible to:

- Set the output to 1 with 1.
- Set the output to 0 with 0.
- Present the status of the next output with NEXT.

Exit finally from this menu by selecting any other menu than POSITION, because POSITIONING means programming of a position.

Changing the status of an output may affect the peripheral equipment which in turn might cause damage or injuries.

**IN/OUT (REG)**

Manual check and possible change of the value in the number register.

Menu: MANUAL

Function: IN/OUT

1. Select IN/OUT under the MANUAL menu.
2. Select REG.
3. Specify register number 0-119.

The value in the register selected is now shown. It is possible to:

- Change the value with CHANGE.
- Present the value in the next register with NEXT.

9.5 TOOL**TOOL (TCP, CHANGE, MAN DEF)**

Manual definition of TCP or BASEP

Menu: MANUAL

Function: TOOL

1. Select TOOL under the MANUAL menu.
2. Select TCP.
3. Specify TCP number 1 -19 for normal TCP. (MH/GL/SW only 20 - 29 are for Room fixed TCP.)
4. Select CHANGE.
5. Select MAN DEF.
6. Normal TCP: Specify x-, y- and z-values in hand coordinates, relative to the turn disc of the robot. If no value is to be changed, press ENTER directly.

Room fixed TCP: x, y and z-values are in the base coordinate system.

Note! Allowed range for x-, y- z- coordinates is $\pm 1500\text{mm}$, and for the vector $\sqrt{X^2 + Y^2 + Z^2}$ for 1KB 2000. $\pm 1500\text{ mm}$, for 1KB 3000 $\pm 1290\text{ mm}$ and for IRB 6000 $\pm 2500\text{ mm}$. If these values are exceeded the error message "TDATA ERROR" is displayed on the programming unit.

7. Activate (see Manual activation of TCP, below) and test run the TCP by manually changing the tool orientation. If necessary, correct the TCP by a new definition, manually or automatically.

TOOL (TCP, CHANGE, AUTO DEF)

Automatic definition of a normal TCP

Facts:

** ^ With the automatic definition it is possible to use an already known TCP value of one tool to measure, with the help of the robot, the TCP coordinates for a new tool. The known "tool" could be the center of the mounting flange, defined by TCP 0.

Procedure:

^^ The known tool is mounted and the corresponding TCP value is activated. Then the robot is positioned with the tool-tip (TCP) in a defined position. This position is stored in the robot memory as the start position.

After the new tool, which should be measured, is mounted and the robot is repositioned so that the TCP of the new tool is in the defined position. This position will be used as the end position, and the system will automatically calculate the new TCP value and store it in the specified register.

If the center of the mounting flange is used as the known TCP, no tool should be mounted when defining the start position, and TCP 0 should be activated.

When the start position has once been defined and stored in the robot memory, it can be reused when measuring new tools, without the need to remounting the old tool and reposition the robot. However, the old TCP must be active when measuring and defining the new TCP.

^

Menu: MANUAL

Function: TOOL

y-s,

1. Activate TCP 0 (see Manual activation of TCP, below).
2. Select TOOL under the MANUAL mode.
3. Select TCP.
4. Specify TCP number 1 -19.
5. If a basepoint is to be defined, press BASEP.
6. Select CHANGE.
7. Select AUTO DEF.
8. If a new starting position for TCP measuring is to be defined, select YES. Otherwise, select NO and go directly to point 9.
9. Dismount the tool, jog the robot manually to a known position, press ST POS and finally, mount the tool again.
10. Jog the robot manually to the known point where the tool is located, and then press ENDPOS.
11. Activate the TCP and test run by orientation of the tool. If necessary, correct the TCP by a new definition automatically or manually.

Automatic definition of a Room fixed TCP.

Move the active normal TCP to the desired position where the Room fixed TCP should be defined.

Press TCP

Enter a value in the range 20 - 29

Press AUTO DEF

Press END POS Use the BREAK function (under point 8 - 9) to, if required, break:

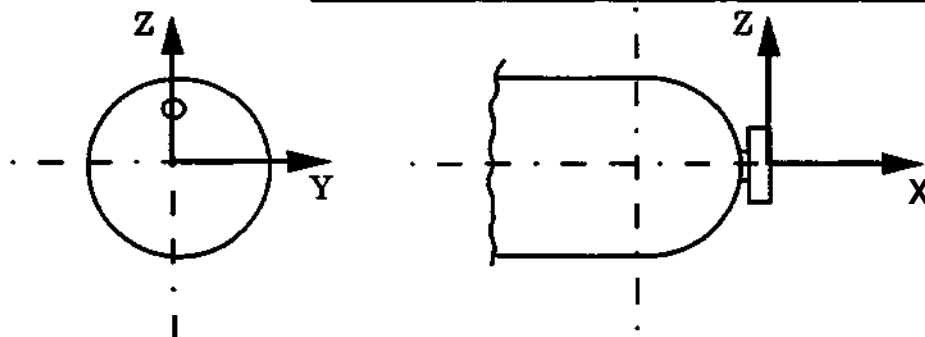
- The entire procedure if the function was selected erroneously.
- Definition of a TCP if the intention is to define a starting position for the TCP definition by running.

For **IRB** 1500, 2000 and 6000 systems, there is a set of TCP aids which simplifies the handling and increases the accuracy when determining the TCP with the function AUTO DEF. The following is a brief description of the operation:

1. The tool, can remain mounted on the mounting flange during the complete procedure.
2. Mount the guide pin in the outer hole of the mounting flange .
3. Define a suitable TCP number for the guide pin at TCP position given according to the table below.
4. Follow the procedure for AUTODEF of TCP in accordance with the programming manual with the following exception:

*) On IRB 1500 it is possible to use the TCP-pin on robots delivered from autumn 1993.

	X	Y	Z	Art.no. for kit
IRB 1500	-6.5	-100	0	3HAA 2246-1
IRB 2000	-8	100	0	3HAA 2246-1
IRB 6000	-15	0	-150	3HAA 0001-UA



- a Activate the previously selected TCP (see point 3 above) instead of TCP 0.
- b When defining the starting point for the TCP measurement the tip of the guide pin (and not the TCP 0) is run to the starting point with ST POS.

TOOL (TCP, ACTIVE)**Manual activation of TCP**

Menu: MANUAL

Function: TOOL

Previous definition of the TCP selected is assumed

1. Select TOOL under the MANUAL menu.
2. Select TCP.
3. Specify TCP number 0 - 29.
4. Select ACTIVE.

The TCP selected is now activated immediately.

**Activation of an incorrect TCP might result in unintentional robot movements.****TOOL (TCP, NEXT)****Control of TCP-position**

Menu: MANUAL

Function: TOOL

1. Select TOOL under the MANUAL menu.
2. Select TCP.
3. Specify the TCP with which the check is to begin.

The x-, y- and z-values, in hand coordinates, of the TCP are now presented.

Room fixed TCP:

The x-, y- and z-values are measured in the base coordinate system.

If the next TCP is to be checked, page with the NEXT function. Finally, when coordinates for a TCP are not available, the TCP concerned has not been defined.

TOOL (TCP, DELETE)

TCP definition canceled

It is assumed that the TCP concerned is not active.

1. Select TOOL under the MANUAL menu.
2. Select TCP.
3. Specify TCP number 1 - 29.
4. Select DELETE.
5. DELETE TCP? YES, NO.

(The coordinates of the TCP selected are now erased immediately.)

TOOL (TCP,BASEP)

Normal TCP only.

It is described below how the tool coordinate system can be defined with a TCP and a Basepoint. First the Startpoint is defined to a reference point in the work piece, for instance with the help of a guide pin as described above, see figure below. After that the tool is mounted and the TCP is defined, see figure below. The Basepoint should now be located on the symmetrical axis of the tool to define the orientation of the tool coordinate system.

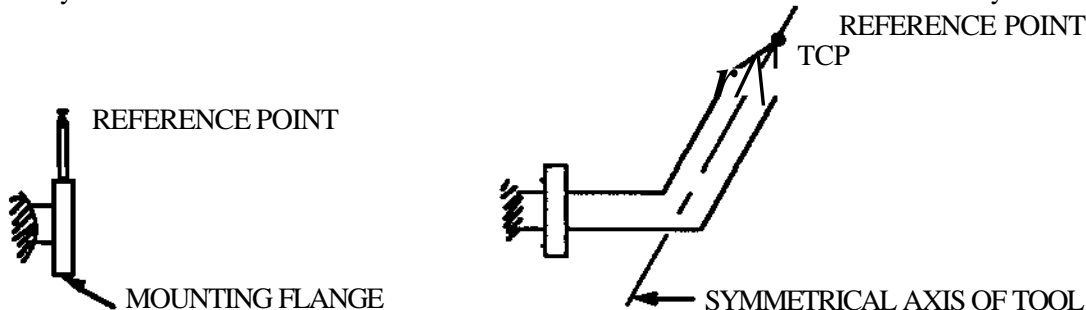


FIG. 7 DEFINING START POSITION

FIG. 8 DEFINING TCP

The base point can be defined manually or by Automatic definition. Below three ways of using the automatic definition are described.

When defining the Basepoint the robot should first be moved to a position where the symmetrical axis or its extension points through the referencepoint, see the figure below. In this figure, the Basepoint will be defined in point B, i.e. outside the TCP.

Note. This means that the z-axis of the tool-coordinate system will point in the opposite tool direction, but this is generally of less importance.

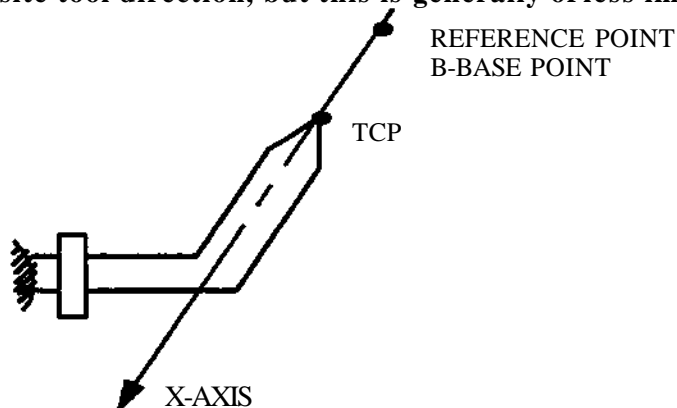


FIG. 9 DEFINING BASEPOINT

Alternative 1:

1. Direct the tool to the orientation required, relative the work piece.
2. Define the TCP with the AUTO DEF method.
3. When ENDPOS is entered the TCP is defined. Move the TCP a couple of centimeters backwards from the work piece by moving the robot manually with the joystick.
4. Define the BASEPoint with the AUTO method without moving the robot, i. e. answer the question "NEW START POS" with NO, and then press ENDPOS.
5. Activate the recently defined TCP.

Alternative 2:

If the tool orientation is to be changed during the program execution, a new pair of basepoint/TCP must be defined.

1. Activate the old TCP.
2. Direct the tool to the orientation required (by manual running).
3. Copy the old TCP-location under the new TCP-number with the AUTO method, i.e. answer the question "NEW START POS" with YES, press START POS and then press ENDPOS, without moving the robot in between.
4. Move the TCP a couple of centimeters backwards from the work piece by moving the robot manually.
5. Define the new basepoint with the AUTO method without moving the robot, i.e. answer the question "NEW START POS" with NO, and then press ENDPOS.

Alternative 3:

Define the TCP (Manually or already defined).

Press BASEP, CHANGE and AUTO DEF.

NEW START POS: answer YES.

Press ST POS

Move the robot in the desired x-direction, showed in the fig. 9.

TOOL (SENSOR)

The following menu for selection of sensor numbers is available under TOOL + SENSOR in the MANUAL menu on the programming unit.

S246				
SENSOR NO =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

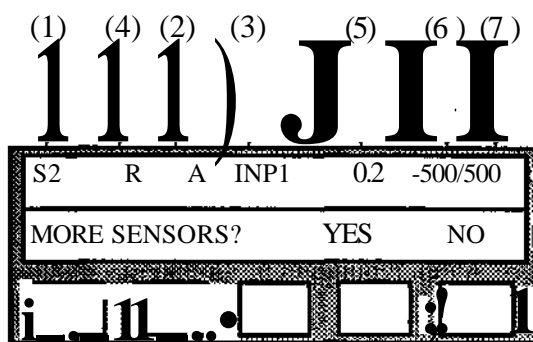
1. The numbers of the sensors which are already defined in the system are presented in the upper display line.
2. Select the number of the sensor which is to be checked, changed or defined.
3. The system then presents a succession of questions for all parameters which are available for the sensor concerned.
4. With ENTER without specification of data, a value already defined for the parameter concerned is not changed. This is used if parameter values are only to be checked.
5. When the sensor definition is completed, the following menu is presented:

S2	B	INP237		
MORE SENSORS ?		YES	NO	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

YES: Return to the basic menu for definition of further sensors. NO: Conclusion.

Sensor data upper display line

Parameter values previously defined for the sensor concerned is shown on the upper display line. The display line is updated after each new entry of parameter values. The following figure shows an example of a completely defined sensor.



For the numbering (1) - (7) acc. to the figure above, refer to parameter I to VII acc. to the following:

1	S2	SENSOR NO = 2
2	A	Analog sensor
2	B	NO OF BITS = 2
3	INP1	INPUT NO = 1
4	R	INSTALLED ON ROBOT
		NOT INSTALLED ON ROBOT
5	0,2	SCALE = 0,2
6	-500	MINVALUE = -500
7	500	MAXVALUE = 500

Parameter definitions

Parameter I The parameter refers to the logical number **of the sensor specified** when programming adaptive instructions.

Menu Basic menu for sensor definition.

Guide text SENSOR NO =

Def.range 1-16

Parameter II The parameter specifies whether an installed **sensor is of the** analog or digital type. For digital sensors, the parameter also defines the number of digital bits for the sensor concerned.

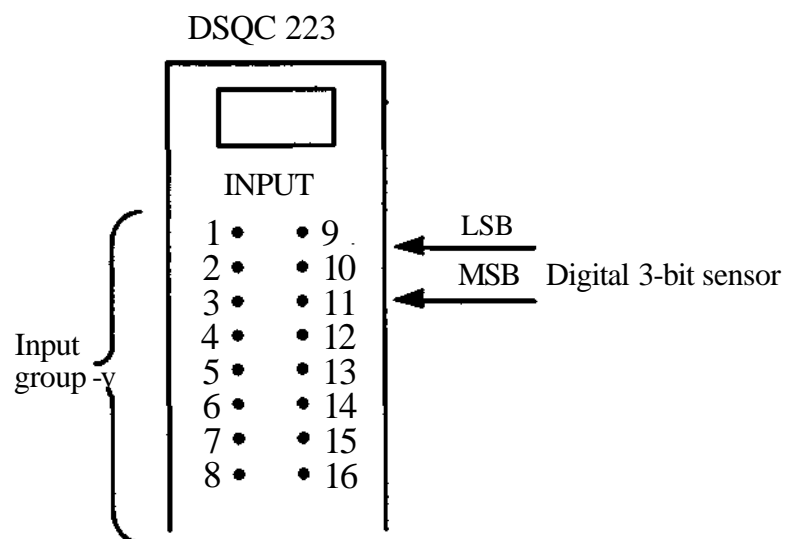
Menu Consequential question above.

Guide text NO OF BITS=

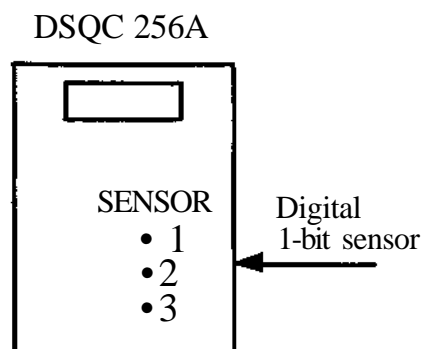
Def.range 0 = Analog sensor.
1 = Digital one bit sensor of ON/OFF type
2 = Digital two bit sensor.
The two bit sensor can be interpreted as a one bit sensor with a sign bit (MSB).

Note It is possible to define a digital multi bit sensor with 3-8 bits, but the performance is unspecified.

Parameter III	The parameter specifies, for an analog or digital sensor, to which physical inputs or outputs the sensor is connected.
Menu	Consequential question above
Guide text	INPUT NO=
Def.range	1-4 = Analog inputs 1-96 = Normal digital inputs 237 - 239 = Fast digital inputs
Type of inputs	<p>1. Analog inputs: The parameter value 1-4 corresponds to the connection of the sensor to a physical input 1-4 on analog I/O board DSQC 209. E.g. Parameter value 2 = Physical input 2.</p> <p>2. Normal digital inputs: The parameter value 1-96 corresponds to the connection of the least significant bit (LSB) of the sensor to logical inputs 1-96 on up to 6 digital I/O board DSQC 223 . For definitions, see section 7.5.3.1. E.g. Parameter value 17 = Logical input 17 i.e. first physical input on the second digital I/O board.</p> <p>3. Fast digital inputs: The parameter values 237-239 correspond to connection of the sensor to sensor input 1-3 on the system board DSQC 256 A. E.g. Parameter value 238 = Sensor input 2.</p>
Note	<p>1. Normal digital inputs:</p> <p>a Intended for digital sensors with 1-2 bits.</p> <p>b The figure below shows how the other bits of the sensor are connected:</p> <p>c A sensor must be connected within the same input group of 8 inputs (SCAN group).</p>



2. Fast digital inputs:
- a Intended for digital one bit sensors for which high speed is required e.g. for searching



Parameter IV	When tracing contours and with free searching, the adaptive procedure is affected by an active correction vector for the sensor concerned. The parameter below defines whether an active correction vector is defined or follows the orientation of the robot wrist.
Menu	Consequential question above "INSTALLED ON ROBOT ?"
Def. range	NO = Active correction vector is fixed in the basic coordinate system of the robot. Corresponding normally to the sensor not being mounted on the robot. YES = Active correction vector is fixed in the robot wrist coordinate system. Means that the correction vector follows any reorientation of the wrist between the programming of the correction vector and the adaptive procedure. Corresponds normally to the sensor being mounted in the hand of the robot.
Parameter V	1. The parameter defines the position gain (correction in mm per bit change of sensor signal) which affects the magnitude of the reference issued by the control system of correction movement with contour tracking. 2. The parameter defines in addition a scale factor which affects the reference issued by the control system for: a The speed of the correction movement with contour tracking, b The search speed with free searching.
Menu	Consequential question above. Final question for digital one bit and two bit sensors.
Guide text	SCALE=
Def.range	0.00 - 99.99
Note	Practically usable values 0,01 - When using contour tracking with weld guard, too high values can result in instabile weaving direction.

9 Manual menu

Parameter VI	The parameter defines the least permissible digital value of the sensor signal i.e. the control system, before use, filters out sensor signals less than a certain value.
Menu	Consequential question above concerning analog sensor or digital (3-8) bit sensor.
Guide text	MINVALUE =
Def.range	For an analog sensor the permissible definition range is ± 1023 . The permissible range corresponds to $\pm (2(n-1)-1)$, where n = the number of bits.
Note	An analog sensor reading of ± 10 V is converted to a digital value of ± 1023 . This corresponds to a sensor with 11 bits of which one is a sign bit.
Parameter VII	The parameter defines the greatest permissible digital value of the sensor signal, i.e. the control signal before use, filters out the sensor signals greater than a certain value.
Menu	Consequential question above. Final question analog sensor or digital (3-8) bit sensor.
Guide text	MAXVALUE =
Def.range	See parameter VI above.
Note	<ol style="list-style-type: none"> 1. See parameter VI above. 2. MAX VALUE and MIN VALUE are to be defined so that the sensor signal receives a suitably large working range.

TOOL (WRIST LOAD)

Manual definition of programmable wrist load (only IRB 6000)

19 sets of wrist loads can be defined (No. 1-19).

Each contains wrist weight, distance in X direction to the centre of gravity of the wrist load from rotary plate of the axis 6, acceleration and position gain indices for axes 4-6. (See chapter 11.9 for detailed description). Wrist load No. 0 can only be changed under the parameter menu as it contains values for normal wrist load.

Any sets of values can be activated manually or during program execution displayed as follows:

The screenshot shows a menu interface for defining wrist loads. The top line displays 'LOAD 0 1* 3 4 7', where the digit '1' is marked with an asterisk to indicate it is the currently active set. Below this, the text 'LOAD NO=' is followed by five empty rectangular input boxes for entering values. To the right of these boxes are two buttons labeled 'CE' and 'ENTER'.

The values in the upper row displays the defined sets of wrist loads, the digit with a star displays the set currently active.

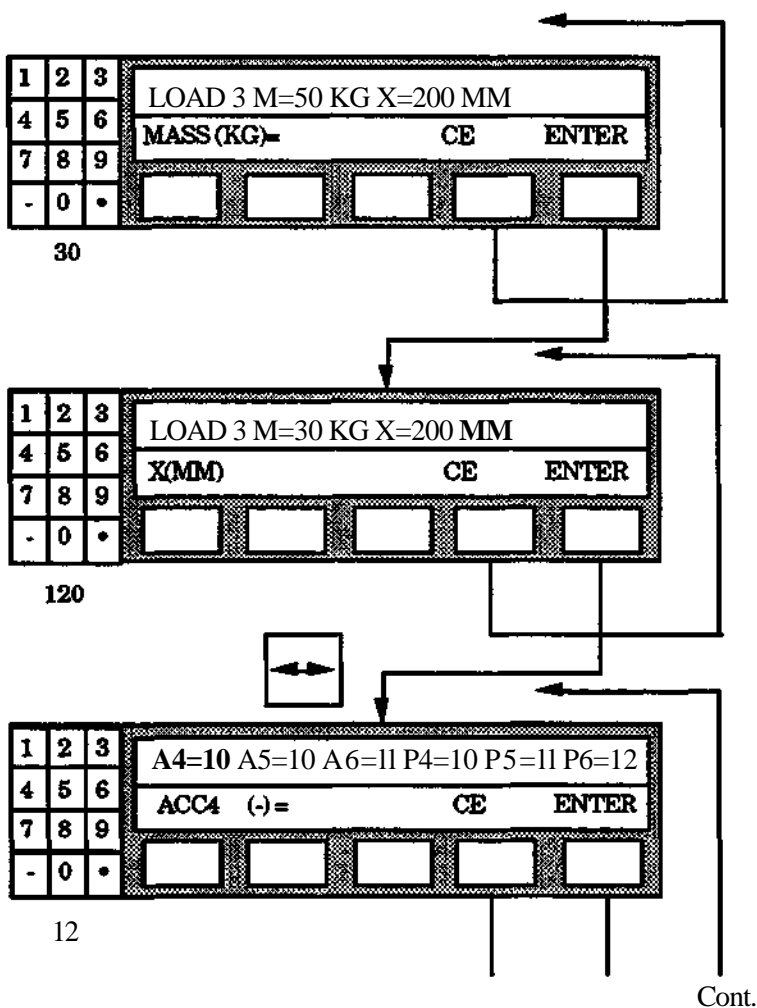
Menu	The following menu will be displayed when WRIST LOAD= 0 -19 is selected.
------	--

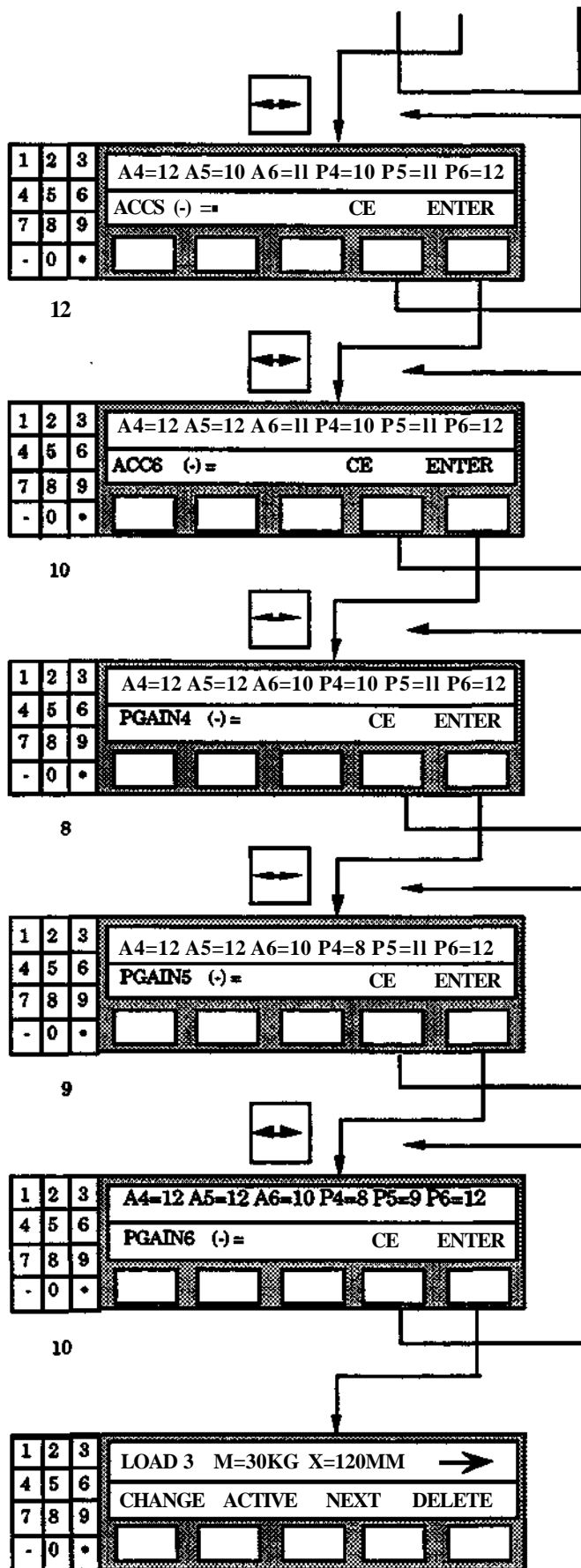
- * Active LOAD
- ** Not for LOAD 0
- *** Not for undefined LOAD
- **** Not for active LOAD

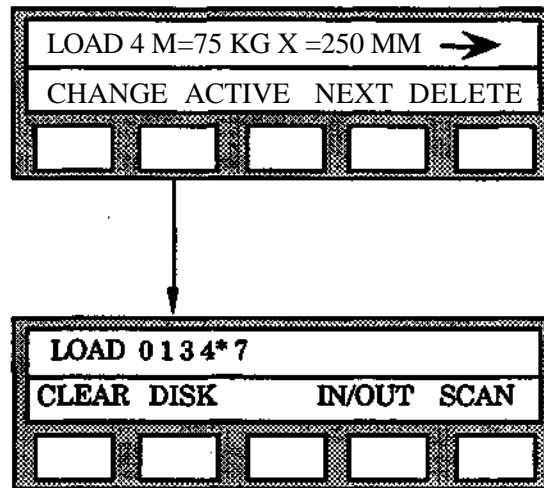
LOAD 3 M=50 KG X=200 MM				
CHANGE**ACTIVE***NEXT DELETE				

When the shift key on the programming unit is pressed the acceleration and position gain indices are displayed.

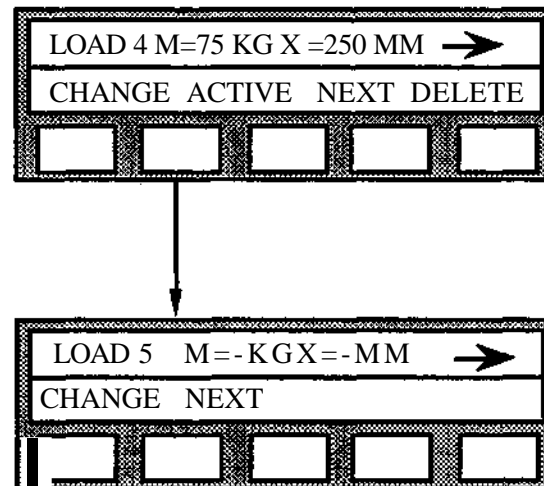
CHANGE





ACTIVE

Current set of WRIST LOAD values is activated.

NEXT

If the WRIST LOAD number is undefined, the values for the acceleration and position gain indices will be updated with predefined values.

DELETE

LOAD 3 M=75 KG X=200 MM →				
CHANGE ACTIVE NEXT DELETE				
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

↓

LOAD3 M= 50 KG X=200 MM →				
DELETE LOAD?		YES	NO	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

TOOL (SOFT S, MH/GL/SW only)**Use of soft servo function**

Nine sets of "softness percentages" can be defined (1-9). Each set contains softness values for each axis. Number 0 can not be changed as it contains values for normal servo control. Number 1- 9 are undefined as default. Any set of values, or normal control, can be activated manually or during program execution displayed as follows:

SOFTSERVO 012*49				
SOFT SERVO		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

The values in the upper row displays the defined sets of softness percentages. The digit with a star displays the set currently active.

Note! If HUB 6000, the current wrist load should be trimmed before defining SOFT SERVO.

MENU

The following menu will be displayed when SOFT SERVO=1-9 is selected and the selected number is active:

SOFT SERVO A1=10 A2=20 A3=30 →				
CHANGE ACTIVE NEXT				DELETE
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

When the shift key on the programming unit is pressed, axes A4-A7 are displayed. Should the 7th axis be undefined it will not be displayed and can not be changed.

A4=15 A5=25 A6=35 A7=45				
CHANGE ACTIVE NEXT				DELETE
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

CHANGE

SOFT SERVO A1=10 A2=20 A3=30 —>•				
A1=		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3
4	5	6
7	8	9

15

SHIFT

SOFTSERVOA1=10 A2=20 A3=30 —>•				
A2=		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



SOFT SERVO A1=10 A2=20 A3=30 —•				
A7=		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

If the SOFT SERVO number was undefined, the values for all axes will be updated with the values for normal servo (0).

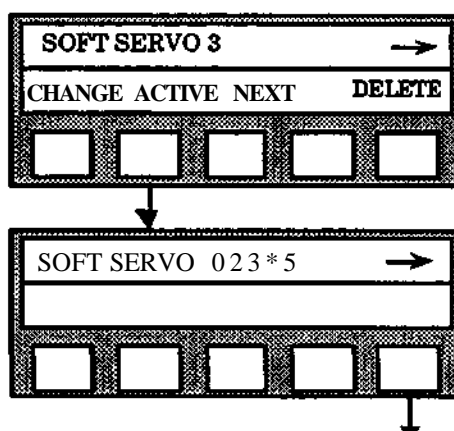
All values entered with the ENTER key will be displayed in the upper row. ENTER without a value will leave the old value displayed.

When the SOFT SERVO number is ACTIVE, all changes will immediately be transferred to the SERVO COMPUTER. Accordingly, no activation of the SOFT SERVO number is necessary after a change if the number was already active.

When the desired value has been entered for axis A7 the display will return to the SOFT SERVO menu.

The values for axes A1-A7 are defined in percent (0-100%). 0% corresponds to normal servo. 1-100% activates SOFT SERVO. Maximum SOFT SERVO is obtained at 100%.

ACTIVE

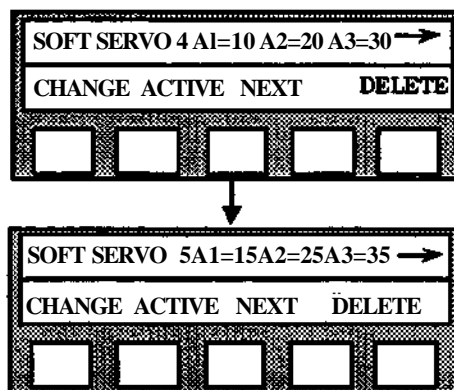


Current set of SOFT SERVO values is activated.

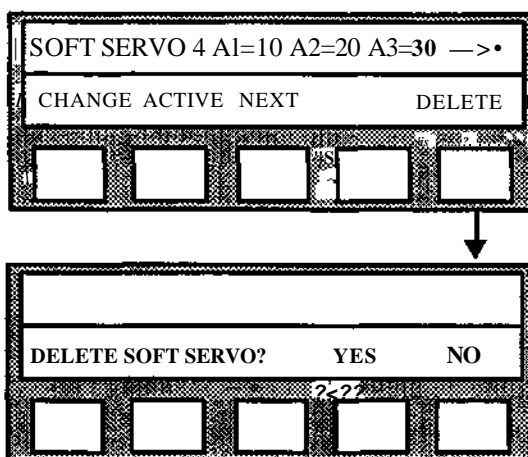


When using the Soft Servo function, the robot speed and path is changed.

NEXT



The next set of SOFT SERVO values is displayed. If NEXT is pressed at SOFT SERVO 9 the display returns to SOFT SERVO 0.

DELETE

YES: Current set of SOFT SERVO values are deleted. The SOFT SERVO number becomes undefined. Return to the SOFT SERVO menu.

NO: Nothing will happen and the SOFT SERVO menu will be displayed.

9.6 FRAME

Menu: MANUAL

Function: FRAME

1. Check that program displacement 0 is active using the SHIFT button, see section 4.5. stored displacements give an uncertain result
2. Select FRAME under the MANUAL menu.
3. Select DEFINE (RESET resets program displacement to its original place.)
4. Type in FRAME number 1 - 5.
5. Run the tool to the original position 1, select FRAME under the MANUAL menu and then press ST POS 1.
6. Position the tool with the joystick to the new position 1 and then press END POS 1.
7. Repeat points 5 - 6 above for position 2 (ST POS 2 and ENDPOS 2 are presented on the display).
8. Repeat points 5 - 6 for position 3 (ST POS 3 and ENDPOS 3 are presented on the display).

Note!

If the operator happens to release the safety pad during jogging, the system exits from the function selected. In that case, start from the beginning once again with function selection.

Automatic definition

The entire procedure is described in detail in section 10.7.

9.7 LIST**9.7. PROGRAM,****Program printout**

Menu: MANUAL

Function: LIST

Means:

Readout of program on a display or printout on a printer (option).

Facts:

A printout queue of up to five programs can be requested and the printouts are performed in the order in which they are commanded. The printout of a program can be commanded while another program is under execution.

The printout is in groups of 60 lines each to suit A4 format paper.

Executed:

The system reacts immediately after the conclusion of the procedure (one program at a time, if there is no queue of printouts to be processed).

Procedure:

1. Ensure that the printer or screen is switched on.
2. Select LIST under the MANUAL menu.
3. Select PROGRAM.
4. Specify the number 0 - 9999 of the program to be printed out.

The procedure is now completed.

Note!

Avoid the following operations during program printout:

- Clearing of program block
- Floppy disk operations
- Manual program selection
- Erasure of programs

The reason is that you can indirectly interfere the program which the system prints out, with the consequence that the printout is faulty.

9.7.2 LIST (ERRORS)

Printout of all error messages in the buffer

Menu: MANUAL

Function: LIST

1. Select LIST under the MANUAL menu
2. Select ERRORS

The system responds by printing out a list of the messages in the buffer. The version of the control program is also printed out.

9.7.3 SYSPAR.

Select LIST under the MANUAL menu

2. Select SYSPAR

The function prints the system parameters i.e. parameters under PARMICHANGE.

9.7.4 USEPAR.

Select LIST under the MANUAL menu

2. Select USEPAR

The function prints user parameters i.e. TCP, SENSOR, EXTFRAME, FRAME, LANG.

9.8 FR SC

Loading of program from SC

The function FROM SC is used to load program blocks or individual programs from the SC. The different procedures are described below.

Note that the remote operation mode must be selected before FROM SC can be used.

Procedure A

Loading of a program block is performed as follows:

1. Select the menu MANUAL.
2. Press function button SCAN twice.
3. Press function button FROM SC.
4. Press function button BLOCK
5. Enter the number (storage identity) of the system data block required (0-9999) with the numerical button set.
6. Press the function button ENTER.

The loading begins immediately after this. The upper line of the display presents the text:

- "COMMAND IN PROGRESS" (during the loading phase itself)
- "READY" (when the loading is complete)

Procedure B

The loading of an individual program is performed as described in the following:

1. Select the menu MANUAL.
2. Press function button SCAN twice.
3. Press function button FROM SC.
4. Press function button PROGRAM.
5. Enter the number (0-9999) of the program block required (in which the program sought is located) with the numerical button set.
6. Press the function button ENTER
7. Enter the number of the program required (storage identity) in the block (0-9999) with the numerical button set.
8. Press function button ENTER.

The entry begins immediately after this. The following text is then presented on the upper line of the programming unit:

- "COMMAND IN PROGRESS" (during the loading phase itself)
- "READY" (when the loading is complete)

9.9 TO SC

Storage of program in SC

The function TO SC is used to store program blocks or individual programs in the SC. The different procedures are described below.

Note that the remote operation mode must be selected before TO SC can be used.

Procedure A

Storage of a program block is performed in accordance with the following:

1. Select the menu MANUAL.
2. Press function button SCAN twice.
3. Press function button TO SC.
4. Press function button BLOCK
5. Enter the number (storage identity) of the program block required (0-9999) with the numerical button set.
6. Press function button ENTER

The storage then begins immediately and when this is completed, the text "READY" is presented on the upper line of the display on the programming unit.

Procedure B

Storage of an individual program is performed in accordance with the following:

1. Select the menu MANUAL.
2. Press function button SCAN twice.
3. Press function button TO SC.
4. Press function button PROGRAM.
5. Enter the number (0-9999) of the program block in which the program sought is to be stored by means of the numerical button set.
6. Press function button ENTER.
7. Enter the number (storage identity) of the program required (0-9999) with the numerical button set.
8. Press the function button ENTER.

The storage then begins immediately and when this is completed, the text "READY" is presented on the upper line of the programming unit display.

9.10 RB MODE

Change of operational mode in relation to SC

The function RB MODE is used when the operational mode of the robot system in relation to the SC is to be changed.

The change is performed in accordance with the following:

1. Select the menu MANUAL.
2. Press function button SCAN twice.
3. Press function button RBMODE.
4. Check which operational mode is currently selected on the upper line of the programming unit display. If a change is to be performed, continue with point 5. Otherwise return to the manual menu by pressing the Manual button.
5. If the remote operation mode is to be selected, press function button REMOTE. If local operation is to be selected, press function button LOCAL.



When the status changes to REMOTE, the robot may be started by an superior computer. Before switching to REMOTE, make sure nobody remains inside the robot working area.

9.11 *LANG

Selection of language

A change of language is obtained in the following manner:

1. Set the robot system in the MOTOR OFF mode.
2. Press the MANUAL button.
3. Scan forward the menu to the text marked with an asterisk at the extreme left.
4. Press the button with the asterisk.i.e the language button.
5. Select a language from those sHown in the menu now presented.

This asterisk is intended to simplify location of the language button in the menus when the language currently in use in the programming unit is not understood by the operator.

9.12 Errors

Display on the programming unit of error messages (after production stop)

Read the error message. For most messages it is possible to get text in plain language by pressing "*" on the programming unit. Use the SHIFT button to display more error messages, if required. If you want to read messages about system errors once again, select errors under the MANUAL meny as below:

Display on the programming unit of error messages from the latest error occasion

Menu: MANUAL

Function: ERRORS

1. Select ERRORS under the MANUAL menu and read the error message.
2. Press "*" to show text in plain language.
3. Use the SHIFT button to display more error messages if required.

Display of error messages on the monitor after production stop

Read the error message displayed on the monitor.

Display of all error messages in the buffer on the monitor

Menu: MANUAL

Function: ERRORS

Select ERRORS under the MANUAL menu and read the error messages that are displayed on the monitor.

Erasure of all messages in the error buffer

Menu: MANUAL

Function: ERRORS

1. Select ERRORS under the MANUAL menu.
2. Select ERASE.
3. Answer with YES. (If the answer is NO the system will not erase any message).

Optional Error message in plain language (to be selected)

Menu: MANUAL

Function: ERRORS

1. Select ERRORS under the MANUAL menu.
2. Select TEXT.
3. Answer with message number and code.

Load texts in plain language into memory

Menu: MANUAL

Function: ERRORS

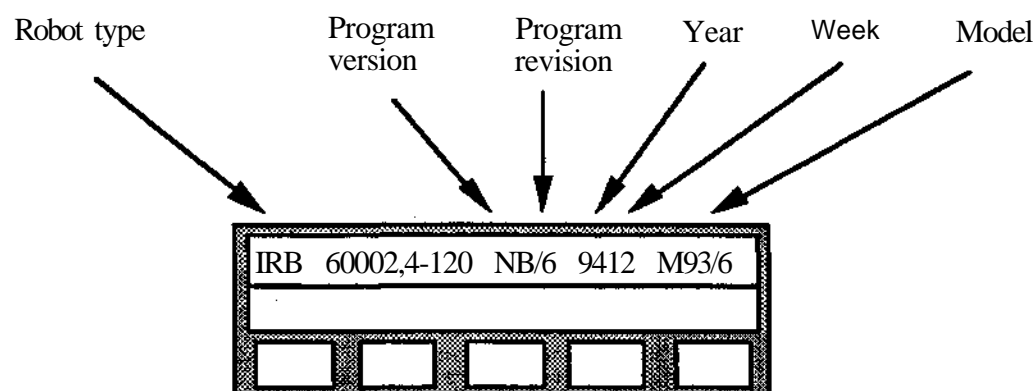
1. Select ERRORS under the MANUAL menu.
2. Select FR DISK
3. Answer with block number (see Installation Manual).

9.12.1 Software Configuration

The controller software configuration can be read by pressing the ERROR button, and then scanning through any errors in the buffer by pressing the SHIFT button until finally the software configuration is shown on the display.

Menu: MANUAL

Function: ERRORS



Display error messages with the SHIFT button until the display shows the following information:

The control program version is also displayed on the monitor.

9.13 TEACH

Position definition for Off-line

Means

Definition of robot positions for use with Off-line programming.

Facts

The TEACH function can only be used when the Off-line Programming Package is connected via a computer link to the robot.

Positioning instructions in programs developed Off-line can refer to robot positions stored on a coordinate file in the development computer.

When TEACH is utilized, the name, position and wrist orientation of each robot position is registered on a coordinate file. When external axes are used, the positions of these are also stored if they are defined in the robot system.

A position is defined by placing the robot in the required position with the joystick and pressing the POS button of the TEACH function.

Used

To be able to use TEACH, the function must start from the Off- line package and be selected under the MANUAL menu on the programming unit.

When TEACH is used in the development computer, all of the TEACH messages and spontaneous messages from the robot system are presented on the development computer display screen. In addition the messages are saved in accordance with the above on the TEACH LOG file in the development computer.

The parameters "Communication board", "Robot identity" and "Computer Link" are to be set correctly in the robot system and the REMOTE mode is to be selected (See Installation S3) for the control system. The parameters "Robot identity" and "Computer link" in the development computer are to have the same values as the corresponding parameters in the robot system.

The robot system is to be in the MOTOR ON mode when TEACH is used.

To check that the connections between the development computer and the robot system function, some action can be taken in the robot system which results in the presentation of a spontaneous message on the computer display. For example, the programming unit can be extracted from the robot system control cabinet and then returned.

Procedure Menu: MANUAL

Function: TEACH

When names on coordinate files and positions are to be specified, the write function on the programming unit is to be used in accordance with the following.

Six letters are shown at each multifunction-button on the lower display line on the programming unit. If for example an "N" is required, the third button from the left is pressed followed by the figure 2 ("N" is the second letter in the group of letters at the third multifunction button). The letter "NT" is then presented on the display. Numbers are typed in at the numerical key set of the programming unit. Characters are erased with the minus sign "-". Each name is concluded with a period sign (.).

9 Manual menu

Sequence definition of positions means that positions are named in sequence when defining. A basic name is given and each new position defined is given the basic name followed by a sequence number. If, for example, the basic name "BOX 1" is selected, the first position defined will be given the designation "BOX 1", the second "BOX 2" etc. The alternative to sequence naming of positions is manual entry of each name.

1 Select TEACH under the menu MANUAL. Then start TEACH from the Off-line package in the development computer.

2 The question "NEW COORDINATE FILE?" is asked.

Answer YES if a new file is to be created and give the name of the new coordinate file.
Answer NO if the coordinate file shown on the upper line of the display is to be used (the coordinate most recently edited).

3 The alternatives DEF, CHANGE, ERASE and EXIT are shown on the lower line of the display.

DEF

4 The system asks if a sequential definition is required.

If YES, give the base name for the position sequence to be defined (e.g. BOX 1). If NO, give the name of the position to be defined (conclude the name with a point).

5 Run the robot to the required position with the joystick and then press the button to define the position.

6A Sequence-definition

The name of the next position (e.g. BOX 2) is shown on the display.

Define the next position in accordance with the point 5 above and repeat the procedure until all positions are defined within the sequence.

Pressing "BREAK" breaks the definition of positions within the sequence and results in the repetition of the presentation on the display of the menu in accordance with point 3 above.

6B Manual entry of each name

The robot system asks if more positions are to be defined.

With YES, give the name of the position and then define the position in accordance with point 5 above. Repeat until all positions have been entered. With NO, the menu in accordance with point 3 above is presented again.

CHANGE

4 The system asks which position is to be changed. Give the name of the position, concluded with a point.

5 Run the robot to the position required with the joystick. Then press the menu button POS to redefine the position.

ERASE

4 The system asks which position is to be erased.

Give the name of the position and conclude with a point. The point named is then erased

EXIT

4 Conclude TEACH.

9.14 WDATA

Weld data (W DATA) contains instructions for defining the different parts of the arcweld process:

- START DATA
- MAIN DATA
- END DATA
- WEAVE DATA
- SENSOR DATA

These are described in chapter 10 under the Arc Welding Process. The instructions are available in the IRB 2000 AW software only.

9.15 PALLET

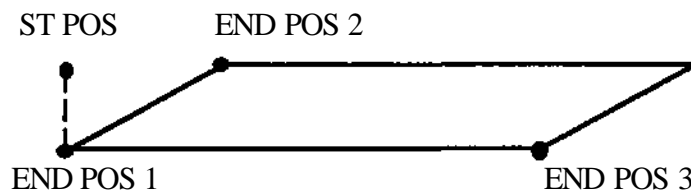
Definition of a pallet. Available on MH/ASM and GLUE software.

Menu: MANUAL

Function: PALLET

- 1 Check that correct TPC is active.
- 2 Select PALLET under the MANUAL menu.
- 3 Specify pallet number 0-9.
- 4 Run the tool with the joystick to the first point in the pallet. Check that the orientation is correct.
- 5 Press END POS1.
- 6 Run the tool to desired position above the pallet.
- 7 Press ST POS
- 8 Run the tool to the last point in the first row.
- 9 Press END POS2
- 10 Specify no of points in every row 1-99.
- 11 Run the tool to the first point in the last row.
- 12 Press END POS3.
- 13 Specify no of rows 1-99

The pallet is now defined. The definition can be interrupted by pressing BREAK. The pallet will then be undefined.



9.16 EXTFRAM (AW only)

Defoliation of EXTFRAM (external frame), see section 10.2.5.5

9.17 HOMEPOS (From M93/5)

HOMEPOS

This function makes it possible to indicate if the robot is in a certain position, the so called HOMEPOS, via setting/resetting of a digital output. Up to three homepositions can be used at the same time.

See also section 7.7 ALIGN/HOMEPOS

The HOMEPOS supervision is done in robot axes coordinates, which means that the HOMEPOS is not a TCP-position, but always refers to a certain mechanical position of all robot axes. As soon as the homeposition is defined, a (selectable) digital output will be set as long as the robot is in (resp. close to) this position.

The zone around the HOMEPOS is +/- 60 increments for each axis. This is the value which corresponds to a zone of 10 mm radius around TCP 0 when the IRB 6000/S3.0 -100 is in it's most extendend position.

The HOMEPOS supervision is done as soon as any home position is defined by the user. The supervision is active during program execution as well as during manual moving via the joystick.

Any defined homepos is internally handled as user parameters like TCP-data and can be stored and loaded to/from disk.

NOTE 1 External axes positions are not included in the HOMEPOS, i.e. it is only the 6 robot axes which are supervised.

NOTE 2 As there is a sampling time of 200 ms and because servo-lags are not observed, the output will - especially at high speeds - not be set at the exact point of time when the robot is in the home position. When using the HOMEPOS-output during program execution - especially at high speeds - special care has to be taken about the timing.

DEFINITION OF THE HOMEPOS AND THE CORRESPONDING OUTPUT

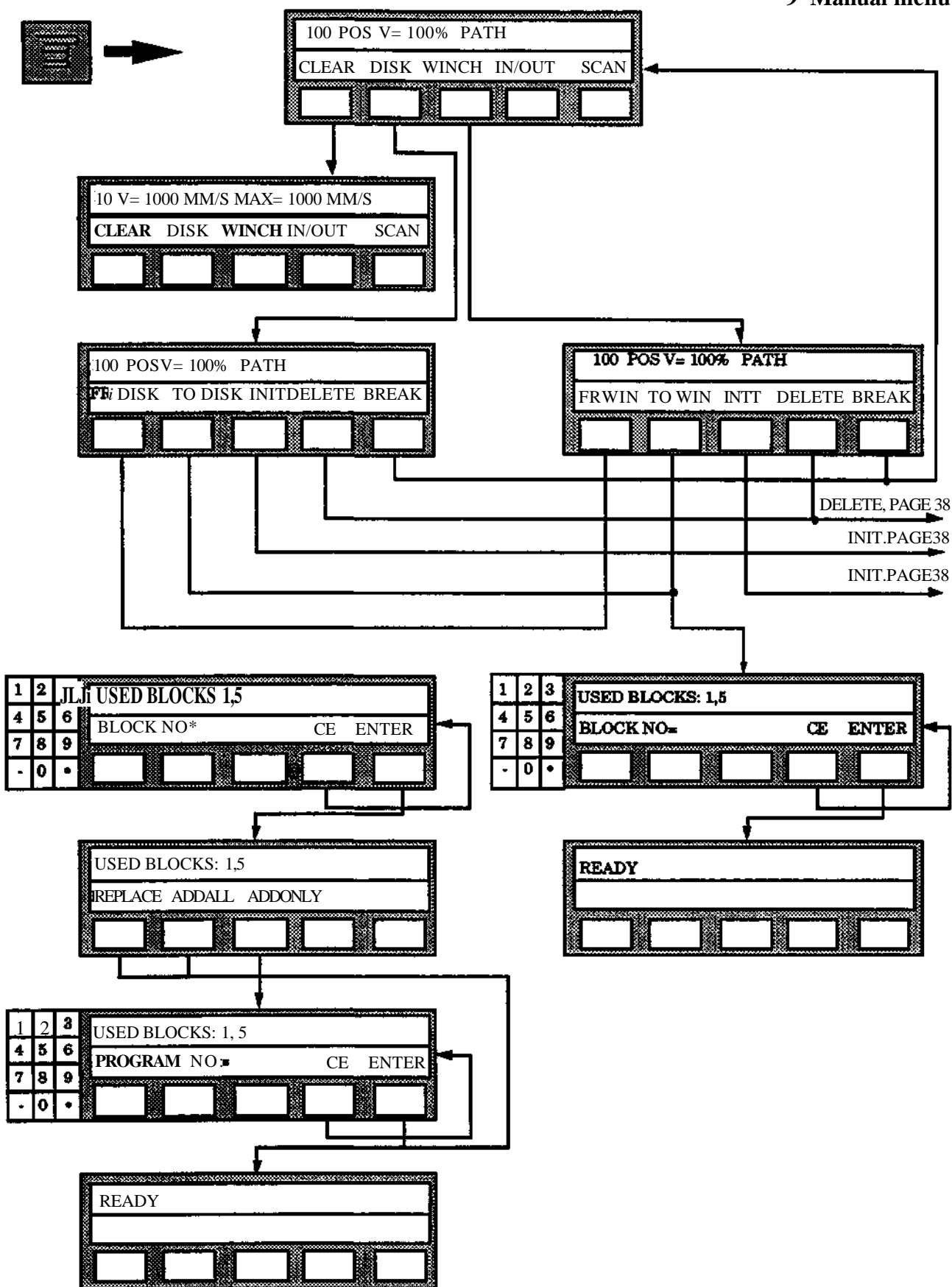
The homepos definition is done under the manual menu.

For a complete homepos definition, the position and a corresponding output have to be defined. The position is defined by moving the robot into the desired position and then enter this position to the HOMEPOS memory. Any free accessible output may be used as homepos indicator.

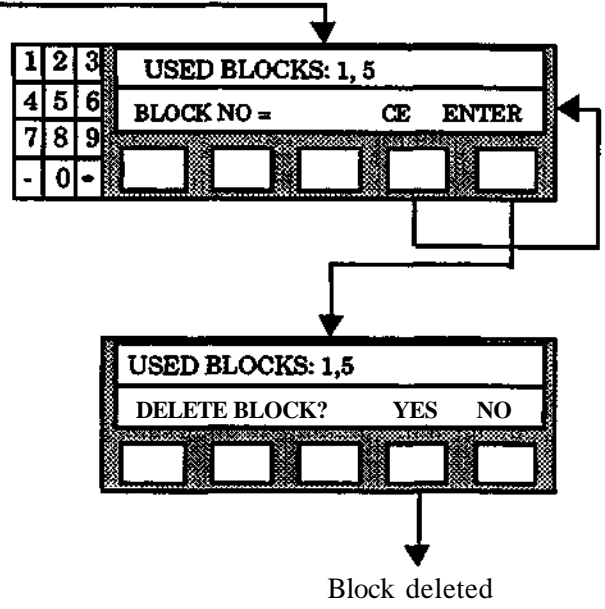
The HOMEPOS definition is done by selecting the HAND menu and then depressing SCAN 4 times.

9.18 REFP

Reference point can be turned off by answering YES.

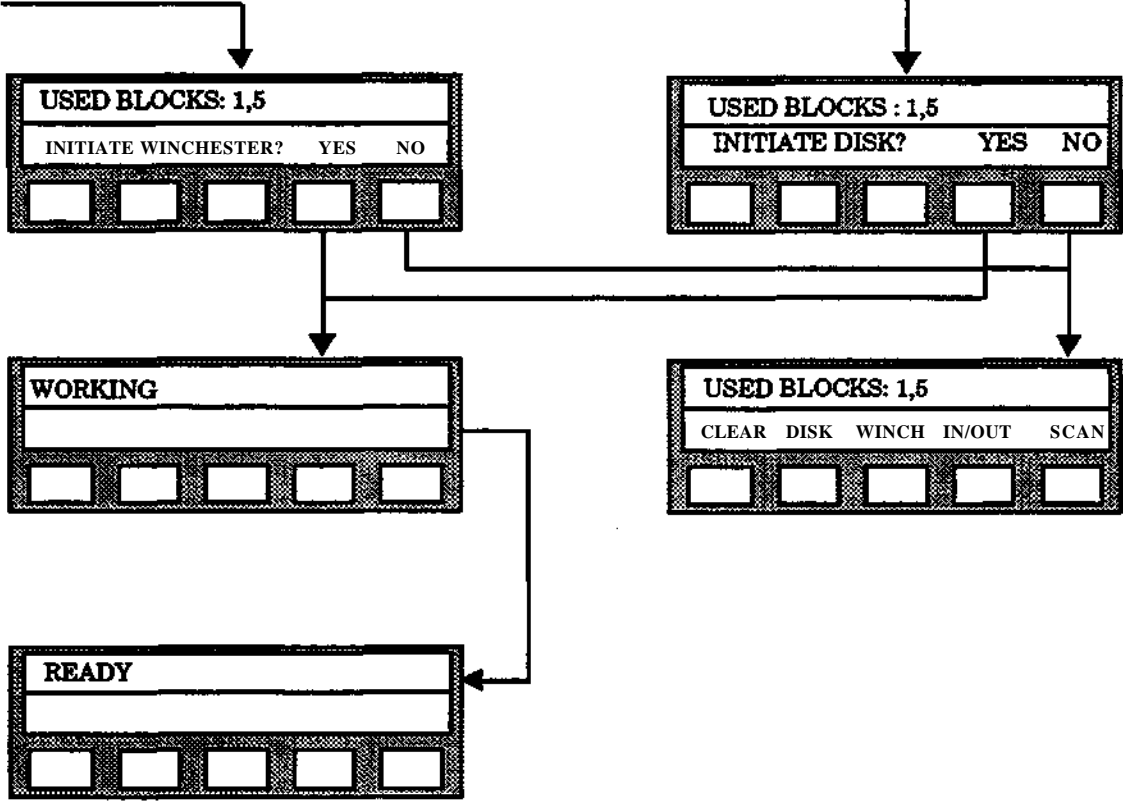


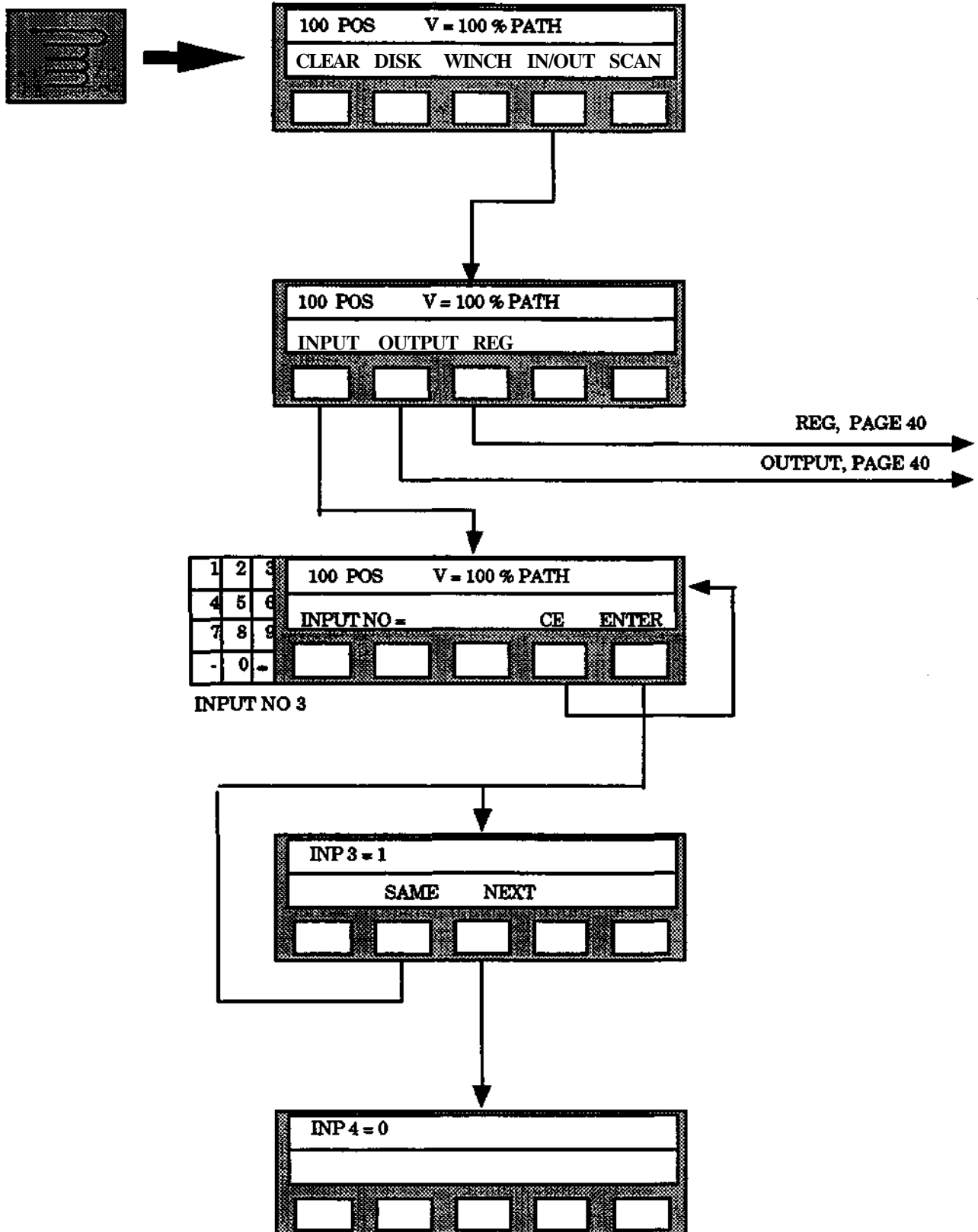
DELETE, PAGE 37



INIT, PAGE 37

INIT, PAGE 37





REG, PAGE 39

OUTPUT, PAGE 39

1	2	3	100 POS		V = 100% PATH	
4	5	6	OUTPUT NO =		CE ENTER	
7	8	9				
-	0	.				

OUTPUT NO 5

1	2	3	100 POS		V = 100% PATH	
4	5	6	REG NO =		CE ENTER	
7	8	9				
-	0	.				

REGISTER NO 11

OUTP 5 = 0					
= 1 = 0 NEXT					

R 11 = 7					
CHANGE NEXT					

OUTP 5 = 1					
= 1 = 0 NEXT					

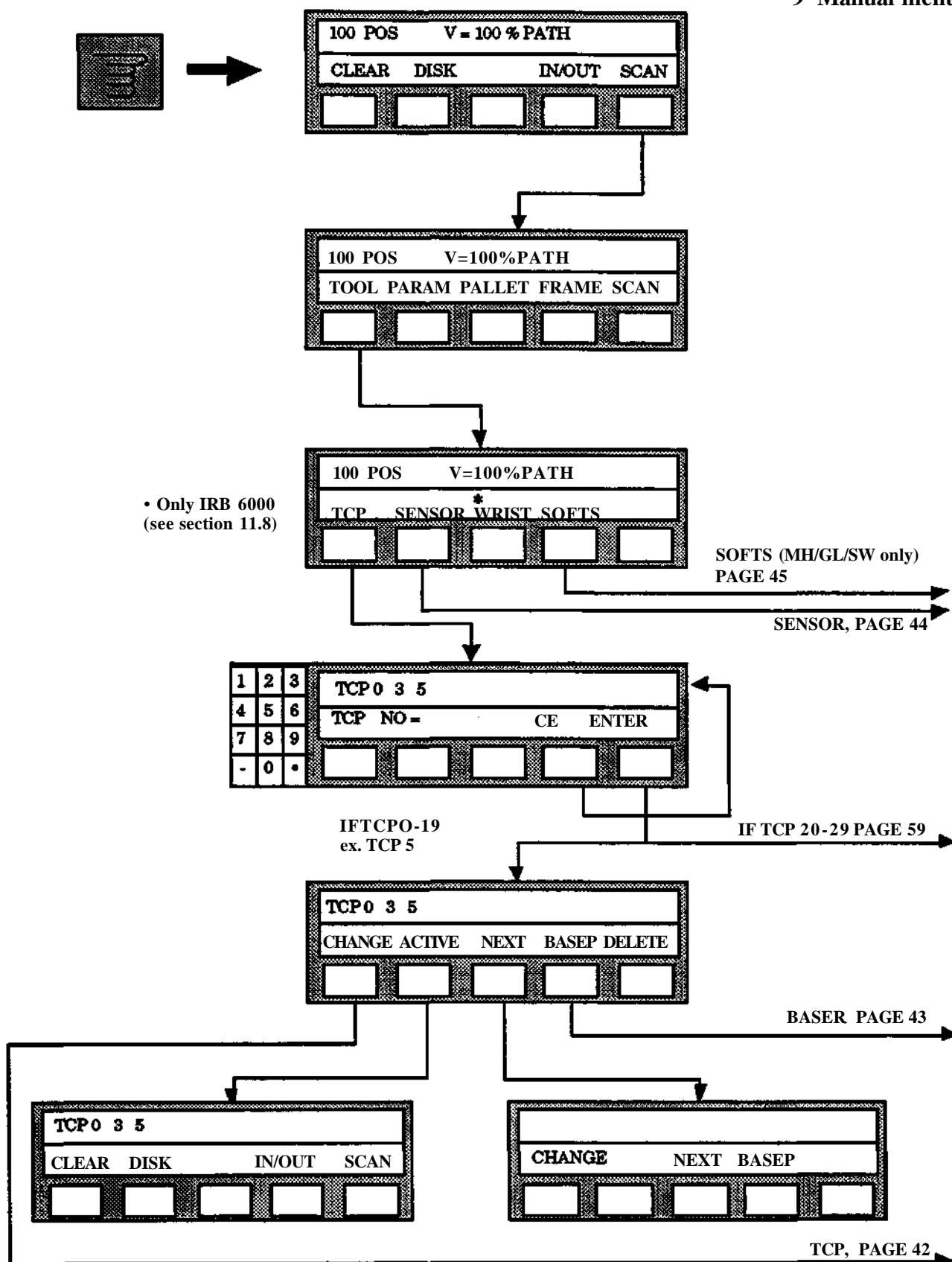
1	2	3	R 11 = 7			
4	5	6	VALUE =		CE ENTER	
7	8	9				
-	0	.				

VALUE 2

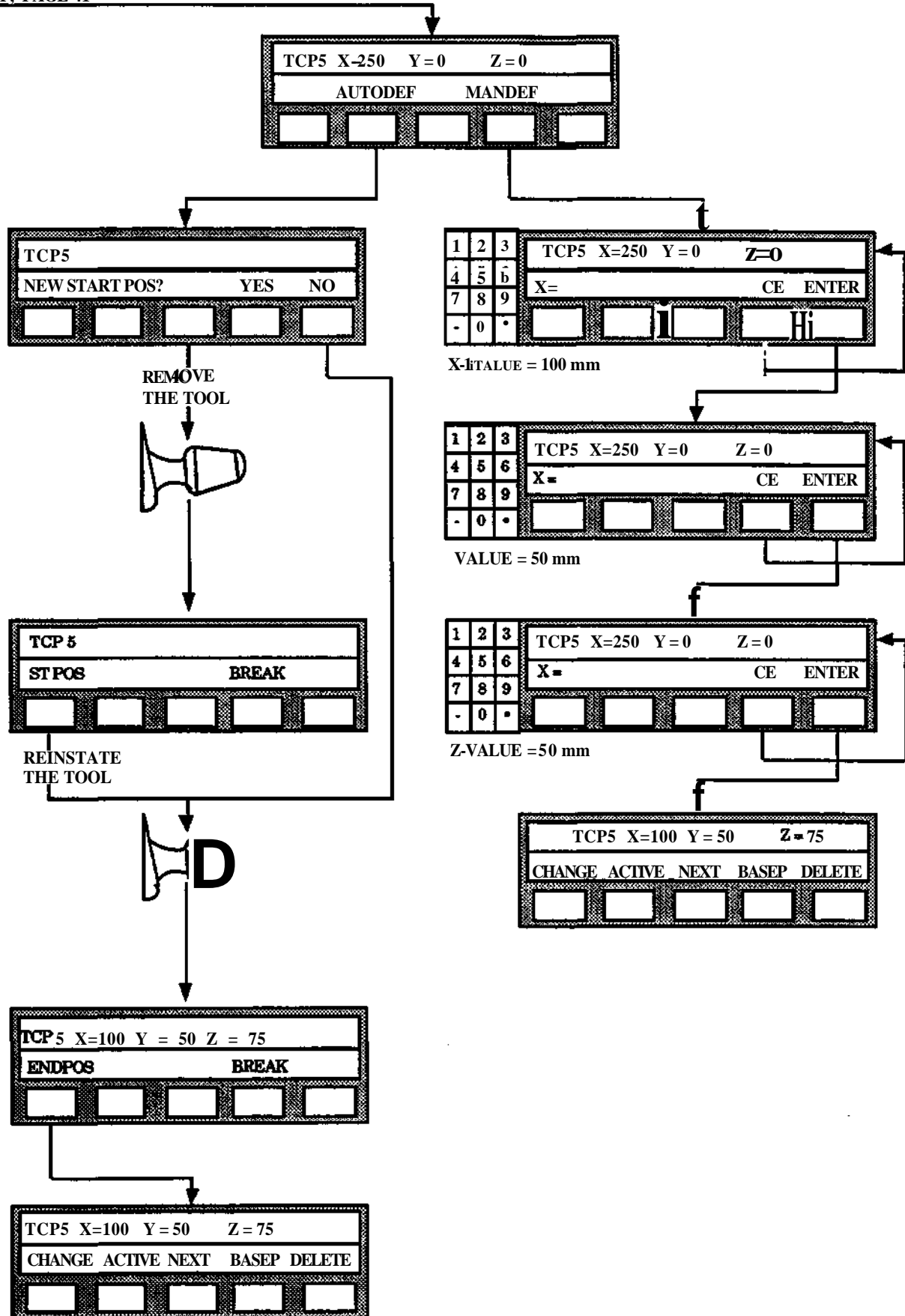
OUTP 5 = 1					
= 1 = 0 NEXT					

R 11 = 2					
CHANGE NEXT					

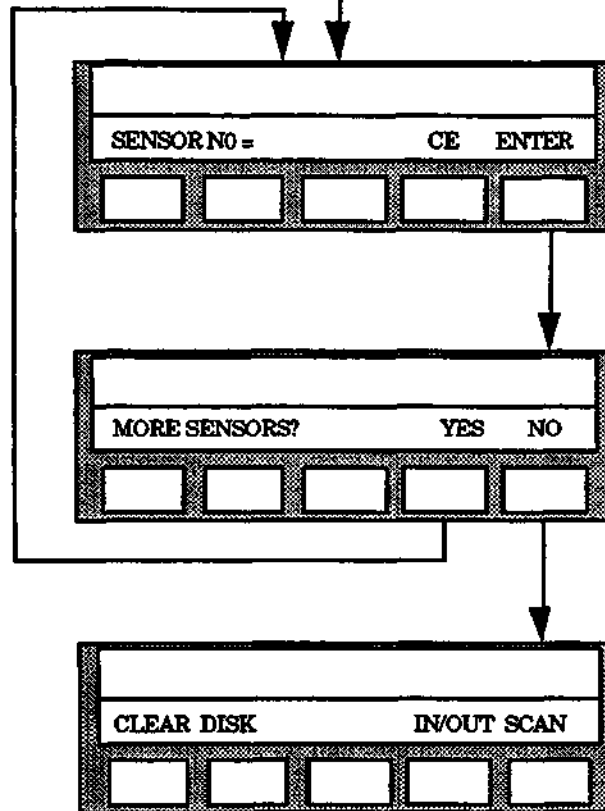
K 11 = 23					
CHANGE NEXT					



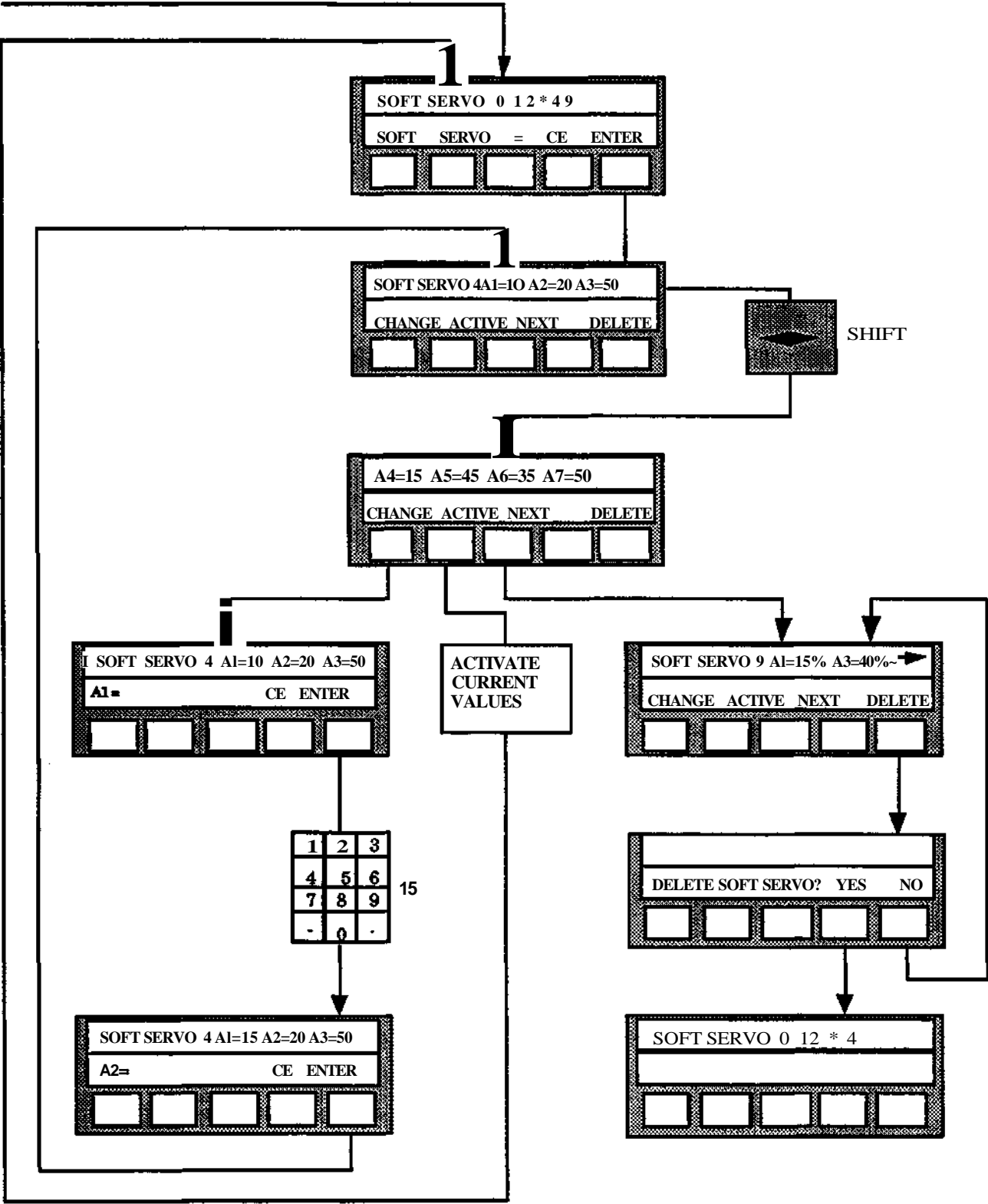
TCP, PAGE 41







See also
SENSOR
INTERFACE
Chapter 10.





100 POS V = 100% PATH				
CLEAR DISK WINCH IN/OUT SCAN				

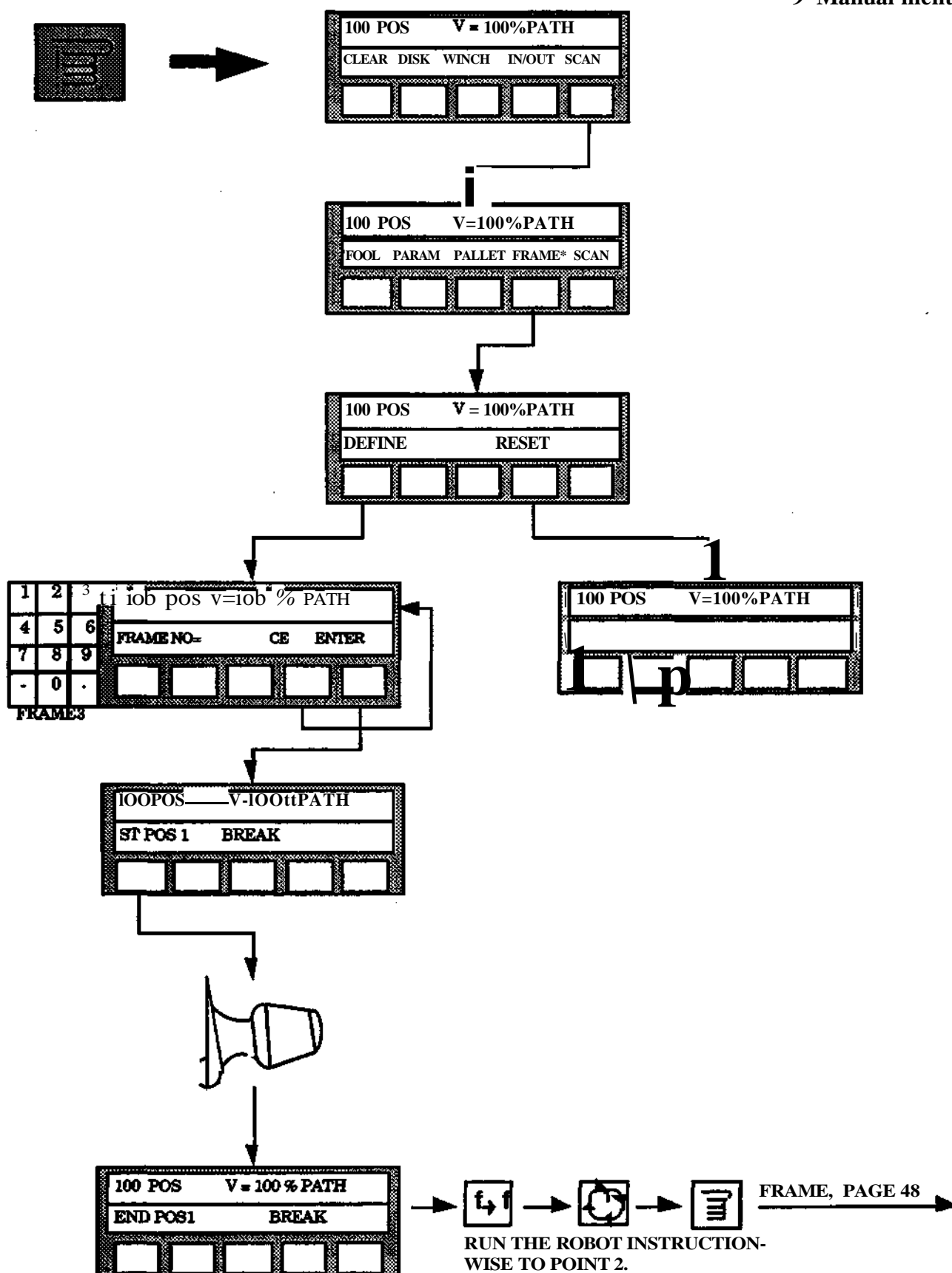


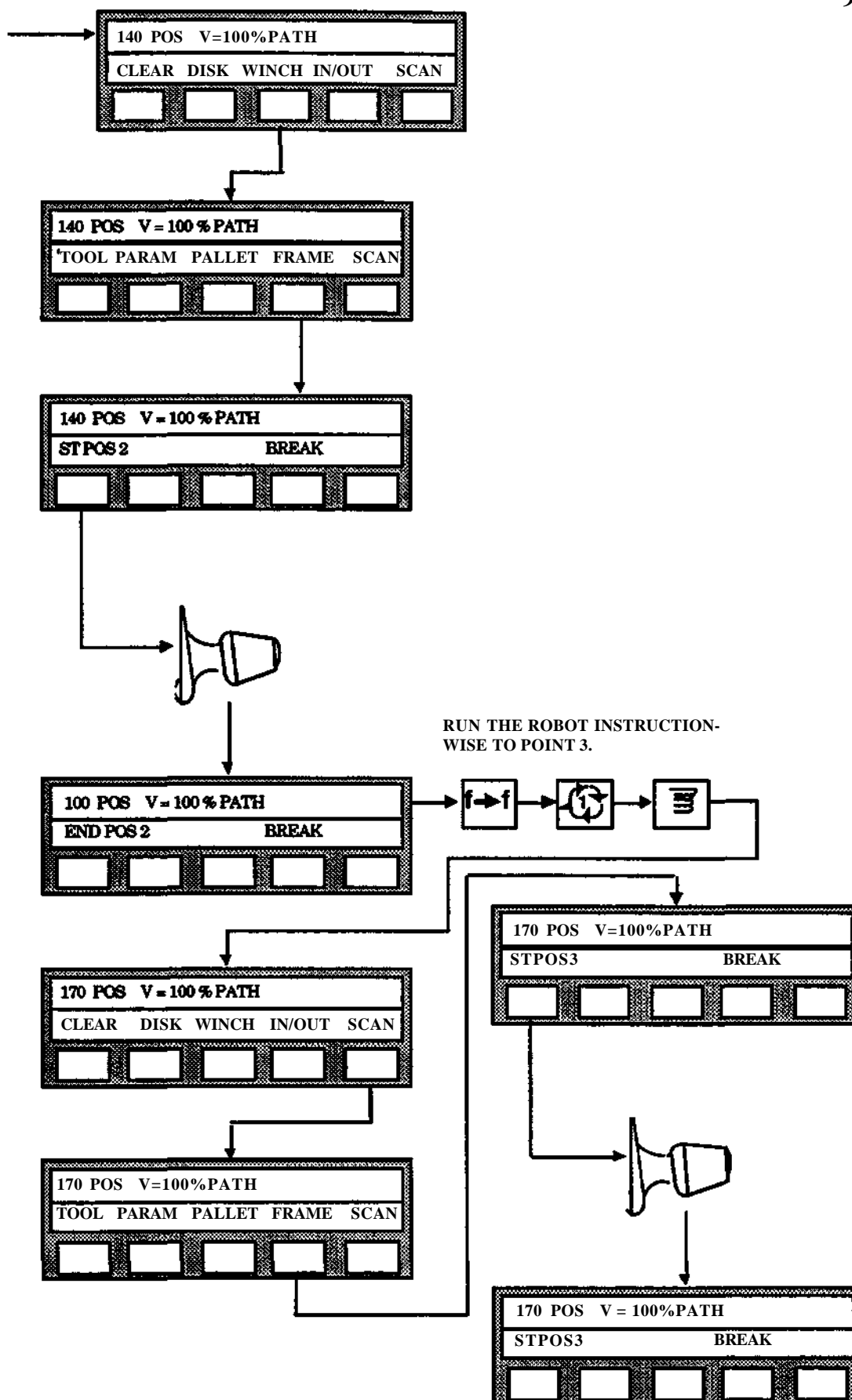
100 POS V = 100% PATH				
TOOL PARAM PALLET FRAME SCAN				

£

— INSTALLA- TION S3









100 POS V=100% PATH				
CLEAR DISK WINCH IN/OUT SCAN				



100 POS V=100% PATH				
POOL PARAM PALLET FRAME SCAN				



*)AVAILABLE ONLY IF THE ROBOT IS PROVIDED WITH PROGRAM PRINTOUT.

100 POS V=100% PATH				
* * * * *				
LIST FRSC TOSC RBMODE SCAN				

**)AVAILABLE ONLY IF THE ROBOT IS PROVIDED WITH COMPUTER LINK.

FROM SC / TO SC. PAGE 50

100 POS V=100% PATH				
PROGRAM ERRORS LTDATAWDATA				

AW SOFTWARE ONLY

AW SOFTWARE WITH LASERTRAK OPTION ONLY

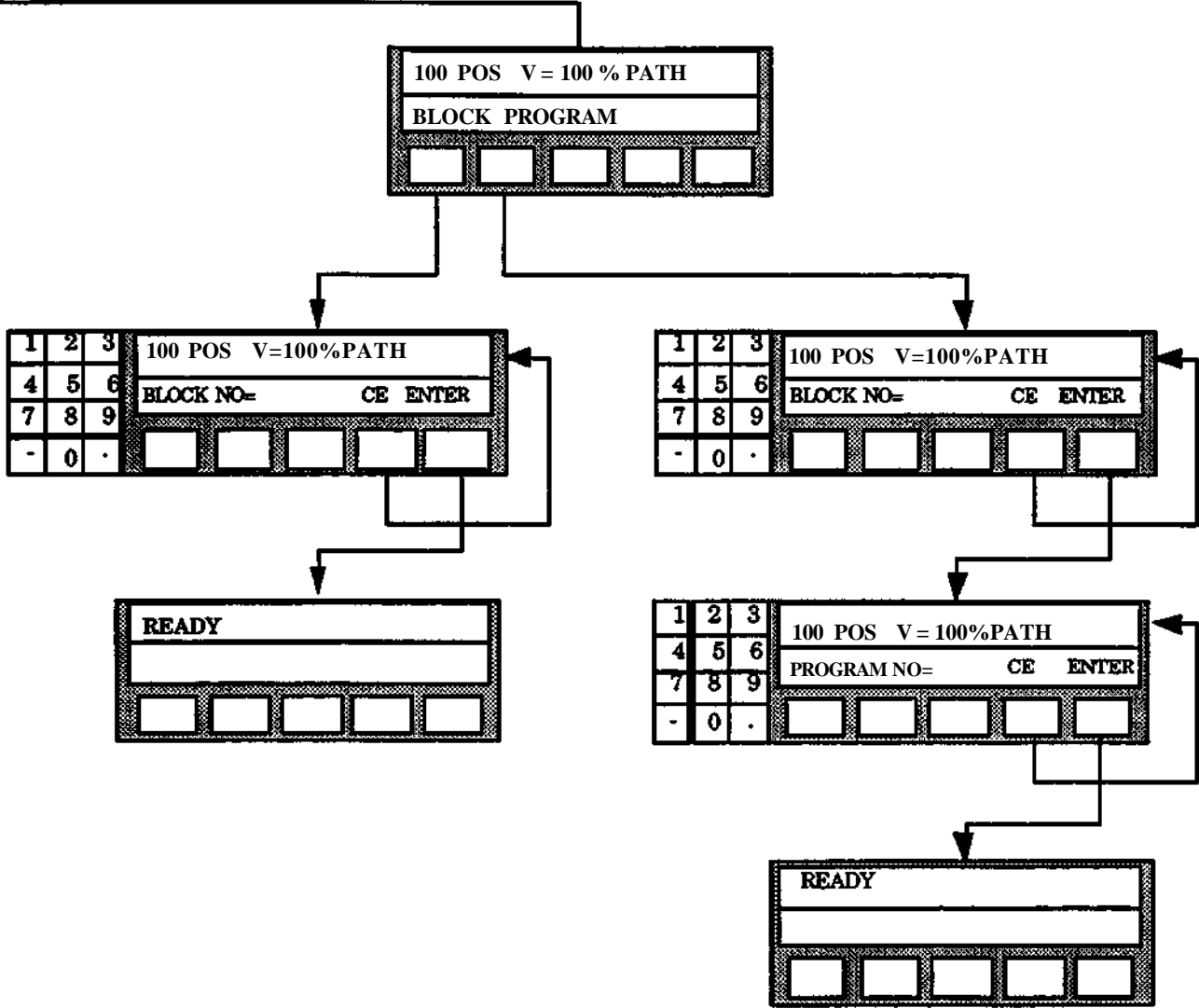
PRINTOUT OF ERROR BUFFER.

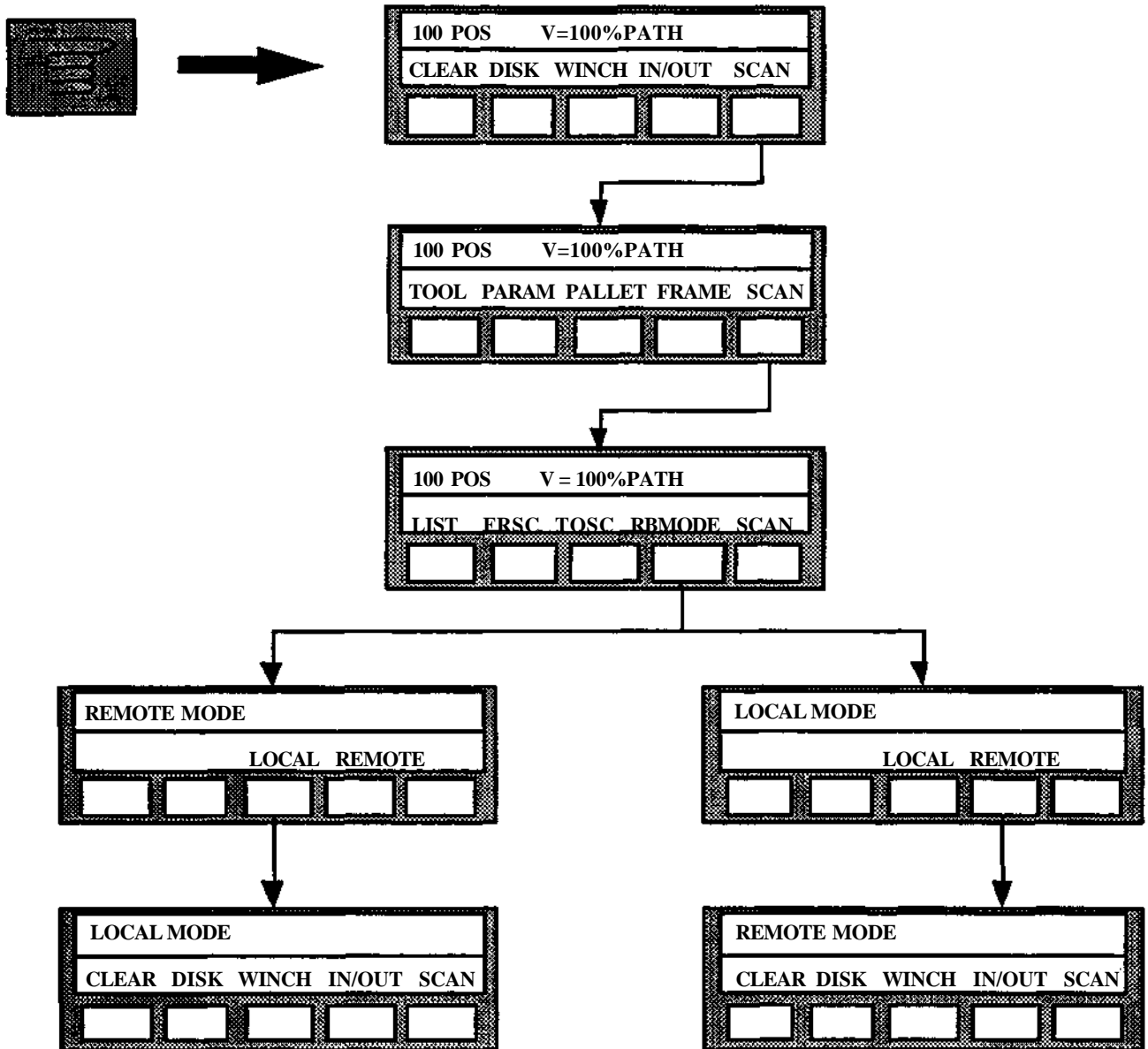
1	2	3	100 POS V=100% PATH	
4	5	6	PROGRAM NO= CE ENTER	
7	8	9		
-	0	.		

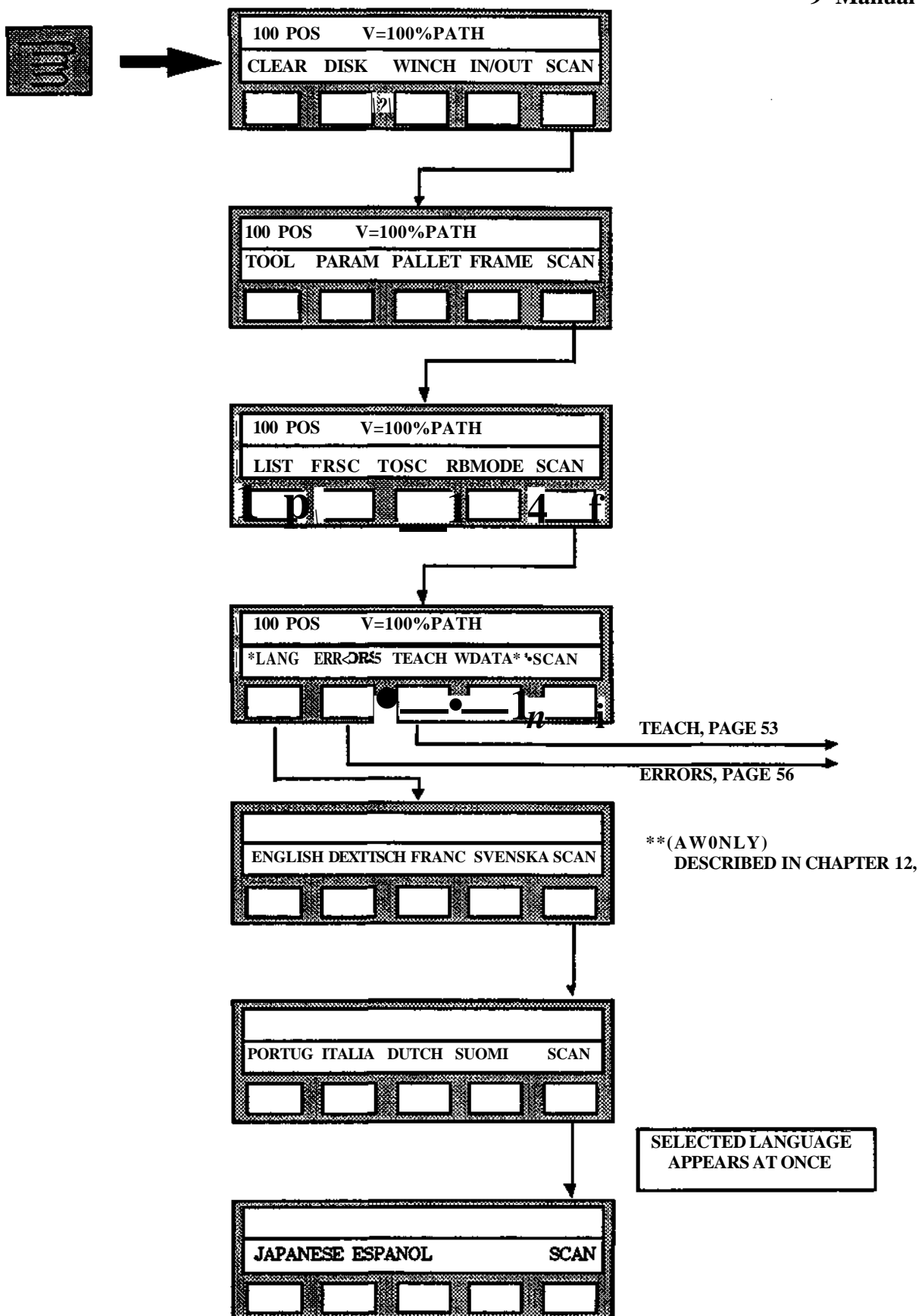


100 POS V=100% PATH				
CLEAR DISK WINCH IN/OUT SCAN				

FROM SC / TO SC. PAGE 49







TEACH, PAGE 52

If no file has been edited previously, the menu below is shown directly.

COORDINATE FILE = AFILE				
NEW COORDINATE FILE? YES NO				

The file edited most recently is shown on the upper line.

1	2	3	COORDINATE FILE = AFILE		
4	5	6	ABCDEF GHUKL MNOFQK STUVWX YZDEF		
7	8	9			
-	0	.			

LETTER NO. 2

COORDINATE FILE * AFILE				
DEF CHANGE ERASE BREAK				

CONCLUDE
TEACH

ERASE, PAGE 55

CHANGE, PAGE 55

DEF, PAGE 54

1	2	3	COORDINATE FILE = T		
4	5	6	ABCDEF GHUKL MNOPQR STUVWX YZ		
7	8	9			
-	0	.			

LETTER NO. 5

1	2	3	COORDINATE FILE - TECHANGE,		
4	5	6	ABCDEF GHUKL MNOPQR STUVWX YZ		
7	8	9			
-	0	.			

LETTER NO. 1

1	2	3	COORDINATE FILE = TES		
4	5	6	ABCDEF GHUKL MNOPQR STUVWX YZ		
7	8	9			
-	0	.			

LETTER NO. 2

1	2	3	COORDINATE FILE = TEST		
4	5	6	ABCDEF GHUKL MNOPQR STUVWX YZ		
7	8	9			
-	0	.			

COORDINATE FILE = TEST				
DEF CHANGE ERASE BREAK				

CONCLUDE
TEACH

ERASE, PAGE 55

CHANGE, PAGE 55

DEF, PAGE 54

DEF, PAGE 53

If no file has been edited previously, the menu below is shown directly.

COORDINATE FILE > ...				
SEQUENCE?			YES	NO

The file edited most recently is shown on the upper line.

1	2	3	NAME;=											
4	5	6	ABCDEF GHIJKL MNOPQR STUVWX YZ											
7	8	9												
-	0	.												

Enter the first name in the sequence with the function and numerical buttons as on the previous page. Conclude with a point

NAME = BOX 1														
POS					BREAK									

Run the robot to the required position for "Box 1" with the joy stick.

NAME = BOX 2														
POS					BREAK									

Run the robot to the required position for "Box 2" with the joy stick.

NAME = BOX 3														
POS					BREAK									

SEE THIS PAGE

CONCLUDE TEACH

1	2	3	NAME =											
4	5	6	ABCDEF GHIJKL MNOPQR STUVWX YZ											
7	8	9	J	I										
-	0	.												

Enter the first name in the sequence with the function and numerical buttons as on the previous page. Conclude with a point.

NAME = SQUARE														
POS														

Run the robot to the required position for "SQUARE" with the joystick.

BOX: X = 30.0 Y = 10.0 Z = 150.5														
MORE POSITIONS?					YES					NO				

SEE THIS PAGE

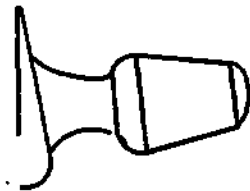
CONCLUDE TEACH

BOX: X = 30.0 Y = 10.0 Z = 150.5																			
DEF					CHANGE					ERASE					BREAK				

ERASE, PAGE 55
CHANGE, PAGE 55

CHANGE, PAGE 53

Enter the name of the position to be changed. Conclude with a point.



Run the robot to the required position for "BOX 2" with the joystick.

1	2	3	NAME =									
4	5	6	ABCDEFGHIJ KLMNOPQR STUVWX YZ									
7	8	9										
-	0	.										

NAME = BOX 2

NAME = BOX 2														
POS														

BOX2: X = 100.2 Y = 25.5 Z = 943.2														
DEF CHANGE ERASE										BREAK				

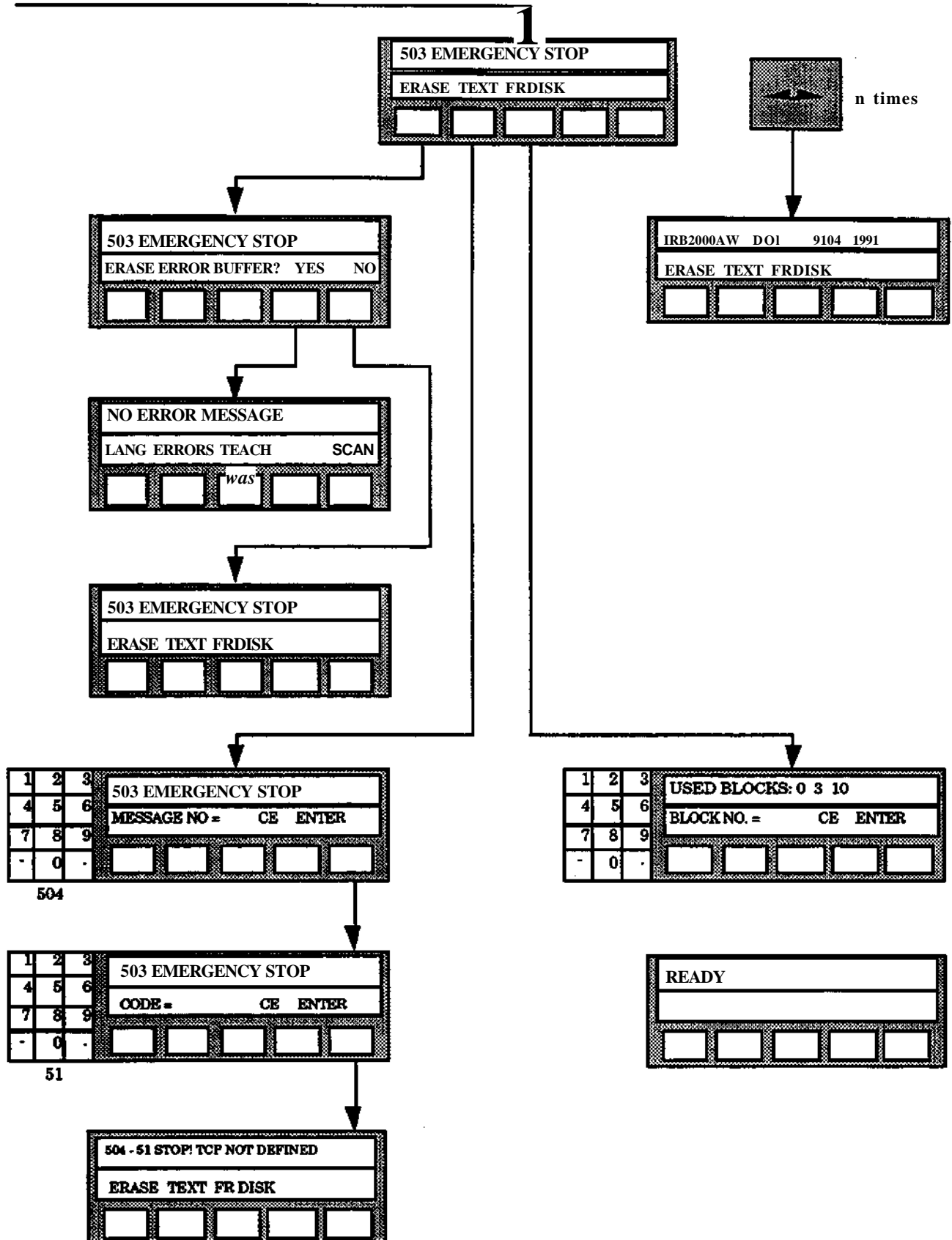
ERASE, PAGE 53

Enter the name of the position which is to be erased. Conclude with a point.

1	2	3	NAME=BOX2									
4	5	6	ABCDEFGHIJ KLMNOPQR STUVWX YZ									
7	8	9										
-	0	.										

NAME = BOX 2

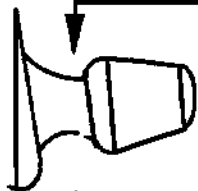
NAME = BOX 2														
DEF CHANGE ERASE										BREAK				



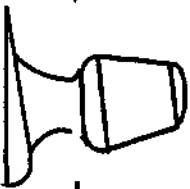
SCAN 1 TIME

PALLET, PAGE 46

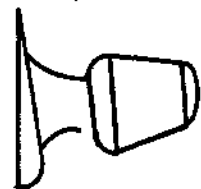
1	2	3	PALLET.....				
4	5	6	PALLETNO =			CE	ENTER
7	8	9					
-	0	.					



PALLET.....							
POS1				BREAK			



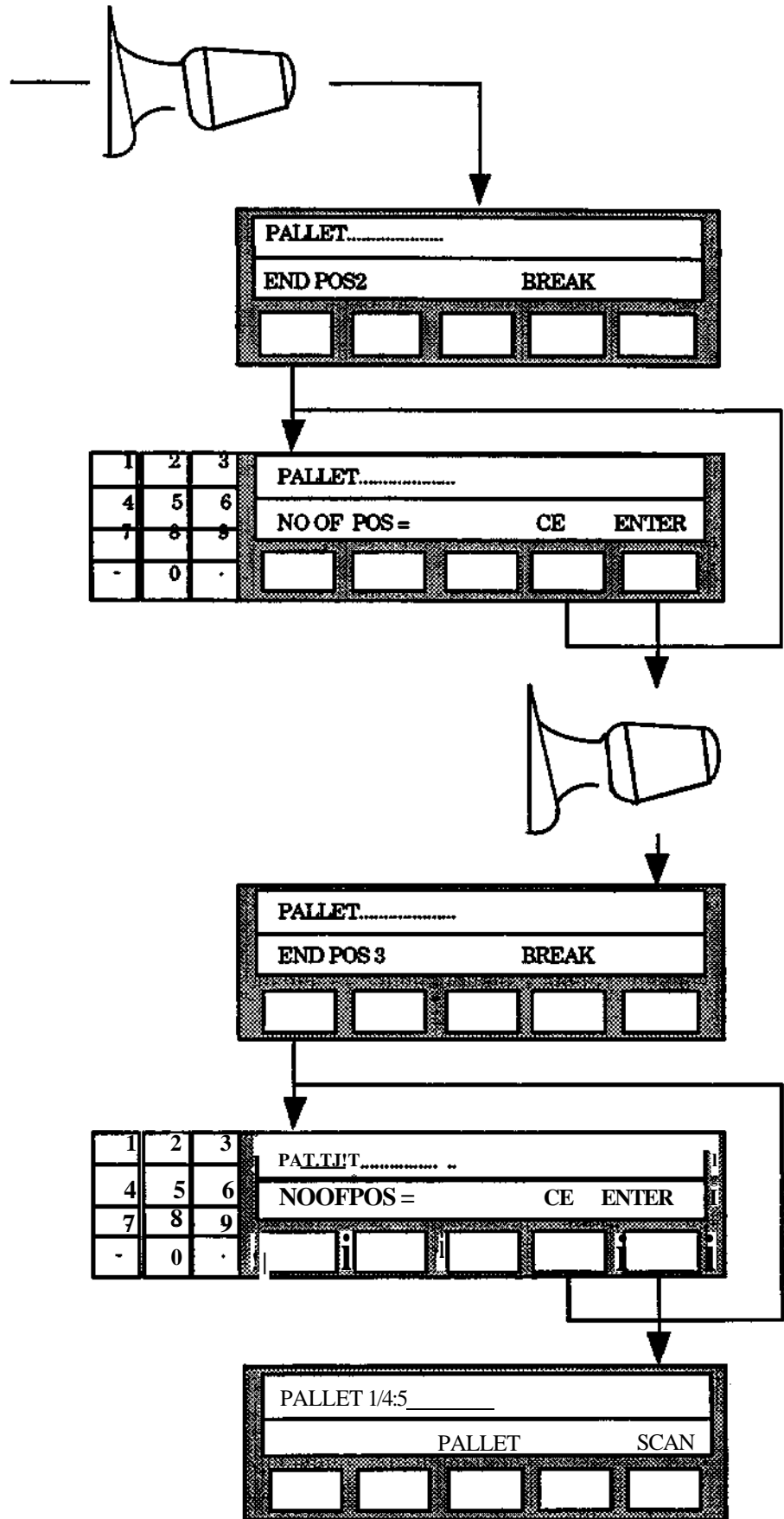
PALLET.....							
ST POS				BREAK			



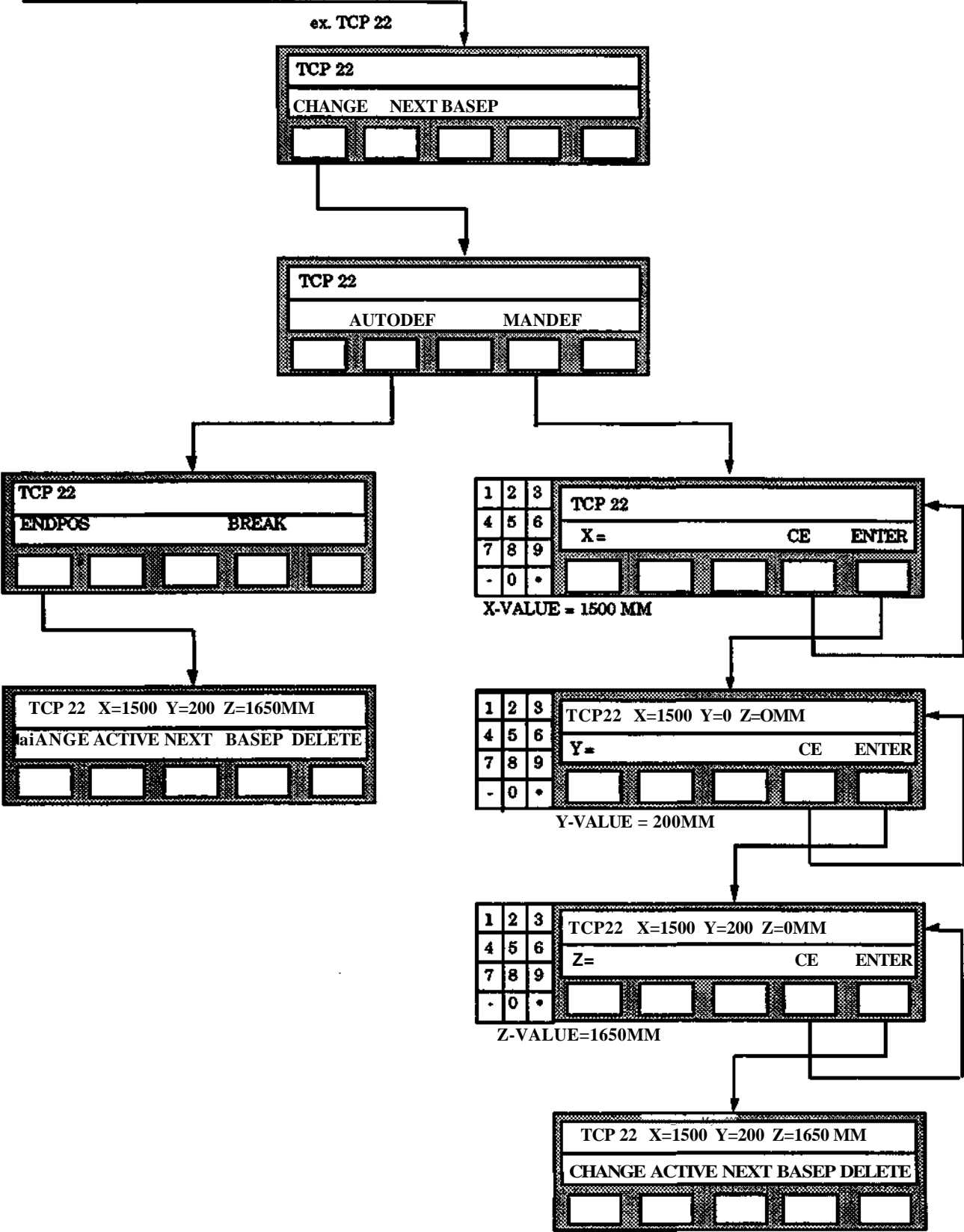
PALLET.....							

PALLET, PAGE 58

FROMSTPOS

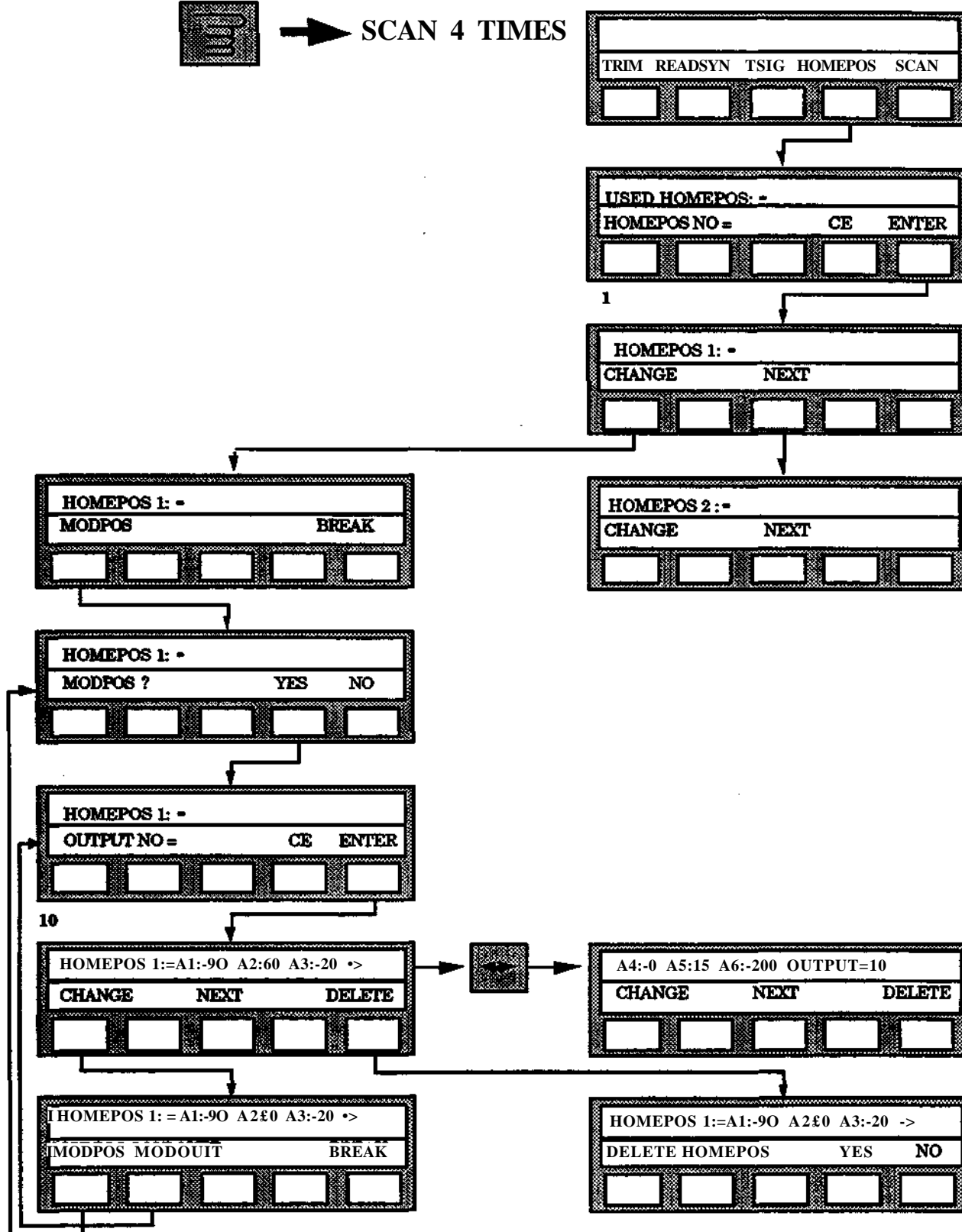


TCP, PAGE 41





➔ SCAN 4 TIMES



10 Adaptivity

Section	Page
10.1 Adaptivity sensor interface	10:3
10.2 Connection and definition of sensors	10:3
10.3 Searching	10:4
10.3.1 Distance searching	
10.3.2 Direction searching	
10.3.3 Autosearch (AW)	
10.4 Speed control	10:7
10.5 Contour tracing	10:8
10.6 Correction vector	10:10
10.7 FRAME	10:11

10.1 Adaptivity sensor interface

Introduction

The adaptivity functions are utilized when it is advantageous to use the same robot program for a task, despite it being possible that the work situation may be changed during the program execution. For example, the task can be to pick up work pieces from a conveyor at standstill, irrespective of the placing of individual units on the conveyor, to apply adhesive to the contour of work pieces each with its individual shape or to grind objects with varying dimensions to a specified dimension.

The adaptivity functions utilize indications from sensors which are connected to the robot system to permit adaption of the robot movements to the position and form of the work piece.

10.2 Connection and definition of sensors

The following sensor types can be connected:

SENSOR TYPE	SIGNAL LEVEL
Digital one bit sensors	High "1"
	Low "0"
Digital two bit sensors	High "01"
	No signal "00"
	Low "10"
	Error status "11" (stop of programmed running)
Analog sensors	-10V to +10V

The sensors can be used for the following functions and are connected in accordance with the table:

SENSOR	FUNCTION	CONNECTED VIA
Digital one bit sensor	Distance searching	System board
	Speed control	System board
Digital two bit sensor	Distance searching	Digital I/O board
	Direction searching	Digital I/O board
	Speed control	Digital I/O board
	Contour tracking	Digital I/O board
Analog sensor	Distance searching	Analog I/O board
	Direction searching	Analog I/O board
	Speed control	Analog I/O board
	Contour tracking	Analog I/O board

A digital sensor can be connected to any digital input on any digital I/O board. Both bits in a two bit sensor are to be connected to input channels within the same supply group. The connections is to be made input channels next to each other, with the lowest bit connection to the input channel with the lower number.

Up to three one-bit sensors for distance searching can also be connected to the direct action inputs of the system board. These inputs have a response time of 12 ± 5 msec, i.e. 7 to 17 msec. The response time for inputs on the digital I/O board $12 (-5 + 15)$ msec,

i.e. 7 to 27 msec.

Connection of one bit sensors via the system board is described in Installation S3, chapter 3.

Analog sensors can be connected to arbitrary analog input.

Except the physical connection according to above, must the sensors be logically defined before the adaptive functions can be used. This is described in chapter 9.

10.3 Searching

The search function is usable for example when the position of the work piece which the robot is to process is not completely known or if it can vary from cycle to cycle.

The robot searches for the part and uses its location as a starting point for the processing.

Two variants of searching can be programmed, distance searching and direction searching. In both cases a search distance is defined during the programming which determines the direction of the searching to be performed.

Up to three sensors can be used simultaneously when searching. When distance searching all sensor types can be used (one bit and two bit digital and analog) and when using direction searching all except digital one bit sensors.

When a sensor "finds" an object in the search distance, its signal level changes. During distance searching, this causes a so-called search stop i.e. the searching movement is interrupted.

During direction searching such a signal change results instead in the start of free searching.

When two-bit sensor are used, a search stop is given when the sensor signal changes level from "01" to "10" (or from "10" to "01"). During free searching under direction searching, a search stop is given when level "00" is searched.

For analog sensors, a signal level is programmed at which a search stop is to be given. The search stop is given irrespective of whether the signal was previously at a level higher or lower than that programmed.

With free searching under direction searching, a search stop is also given when a certain level programmed previously is reached.

As the search stop status is presented at the robot outputs, this can be utilized to control peripherals or as jump conditions etc.

The length of the search movements are determined by the positioning tolerances of the workpiece. To ensure constant search speed the start and stop positions of the searches should be placed approx. 5 mm outside the tolerance range of the workpiece.

10 Adaptivity

If the position of the workpiece varies within very wide limits, the long search movements necessary become excessively time-consuming. An introductory quick search with reduced accuracy can then be used to advantage to give a start point for a search with normal speed and accuracy. To secure that the whole distance of the programmed search movement is executed, the start and end position of the search must be fine points.

Reference point

Reference point instructions are used to refer the search and TCP movements to the preceding search stop. This procedure permits transposition of the TCP movements to the correct position. Reference point instruction REFP should be programmed at a FINE position.

When a reference point is programmed, the current position of the robot TCP is stored in the robot memory. When the reference point instruction is executed, the difference between the stored position when programming and the current position is calculated. This difference is added, with further programming or program execution, to the position of all position instructions until a new reference point is activated or the instruction "REFP OFF" is executed. A new reference point can be activated without deactivation of the preceding reference point.

Conditional jumps

When executing a search movement, the register "Search" is activated to indicate that the search has resulted in a search stop. The search register is one of the arguments with conditional jumps. This permits repetition of unsuccessful searches or jump to an error, control program. If also a jump, conditional with respect to the contents of a register is used, unsuccessful searches can be repeated a programmed number of times before the error control program is called.

10.3.1 Distance searching

If a sensor detects that which is required during a distance searching the searching is interrupted and a search stop is activated. The next instruction in the program is then executed. The search stop applies until a new search begins.

If no search stop is activated, the robot continues to the end of the search distance and then begins execution of the next instruction.

It is also possible to program a delay so that the search stop is not actuated until 0.5 seconds after the sensor detection.

10.3. Direction searching

Direction searching can be used when the robot is to locate the corner of a sheet or a box. When such a corner is located in the search distance, searching continues in a new direction (free searching) without a search stop.

The direction of the free searching is determined by correction vectors, the magnitude and direction of which are programmed with the help of the VECTOR function. (See section 10.6 Correction vector for a more detailed description of how the function is utilized).

The free searching continues until all of the sensors used give signals that no more correction is necessary, (i.e. level "00" for two bit sensors. For analog sensors, the level to apply is selected at programming). The position of the corner is determined and a search stop becomes effective. The search stop can also be delayed until 0.5 seconds after location of the corner.

The stop position is adjusted with respect to the zero zone after which execution of the next instruction is begun.

If no sensor reacts during the first search distance free searching begins at the end of the search distance.

Free searching is stopped if it reaches outside a radius of 5 cm from where it began.

10.3.3 Autosearch (AW only)

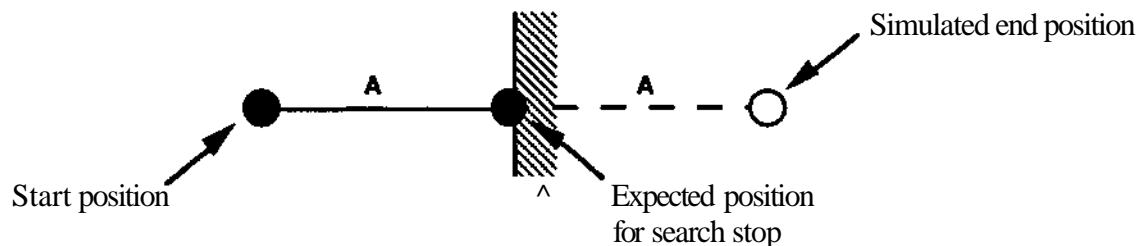
The function AUTOSEARCH is a further development of distance searching. Some differences are described below:

During programming of the search distance, there are sometimes problems to define the end position for the search, depending on difficulties to reach the position physically, or other obstacles. Then it is possible to define a calculated end position, as follows:

- Program the start position of the search in the same way as for distance search.
- * Then program an end position close to the expected position for search stop.

With the instruction AUTO (under the SEARCH menu) it is possible to simulate an end position. It is placed as follows:

1. The operator runs the robot tool to a position, where a search stop is likely to occur. That position is situated on the distance A from the start position for the search.



2. The operator performs the procedure as below.
3. The system will, as an answer, an end position situated at a distance $2 * A$ from the start position for the search.

When the search stop occurs the search is interrupted and the robot position stored. The reference point is also activated (REFP ON). This will simplify complicated searching sequences with many consequent searches.

10.4 Speed control

In some robot applications, the possibility of controlling the working speed of the robot is to advantage, particularly when different parts may require different processing.

A sensor is utilized for speed control so that the sensor signal controls the reduction of the programmed speed. The reduction (X % below) is specified when programming digital sensors. In the case of analog sensors a threshold value and a maximum value are specified (see below)

One bit sensors:

O=Unreduced speed

I=Speed reduced by X %

Two bit sensor:

01=Unreduced speed

00=Speed reduced by X %

10=Speed reduced by 2 x X %

11=Stop

Analog sensor:

When the signal is lower than the threshold value, the speed is not reduced. For signals equal to or greater than the threshold value but less than the maximum value, the speed is reduced in proportion to the signal with signals equal to or greater than the maximum value, the robot movement is halted.

Only one sensor is used with speed control. All types of sensor can be used.

The present velocity correction is saved when the program is stopped (irrespective of how). It is still active when the program restarts (even at backwards execution). The correction is erased first when a POS-instruction without a VCTRL-argument is executed.

10.5 Contour tracing

When the robot is to follow a contour between two points the contour tracing function can be utilized. The function is activated with the argument **CONTOUR**. Start and stop points for the tracing distance are specified during the program as **FINE** positions. During movement between the two points, the sensors signal states where the robot is, in relation to the followed contour.

The position of the robot is corrected if it departs from its path. The direction in which the correction is to be made is given with the help of correction vectors (see section 10.6).

Up to three sensors can be used at the same time when tracing contours. Analog sensors, digital two bit sensors but not one bit sensors can be used.

When programming, the signal level to be adopted by the analog sensors when the contour is followed correctly i.e. when no correction is necessary (a so- called **BIAS** level). For two bit sensors the level "00" is the **BIAS** level.

For contour tracing, the following applies:

- Up to three sensors can be used
- Digital one-bit sensors cannot be used
- Correction vectors for each sensor used must have been programmed previously.

Principle

The contouring function is designed to enable small path corrections to be made when the robot moves between two pre-programmed points. Correction are performed by each interpolated interval (each 48 ms) that are proportional to value of the sensor and the programmed speed of the correction vector.

The actual formula for calculating the correction is as follows:

$$\text{CORR} = (\text{SENSOR VALUE}) * \text{SCALE FACTOR} * (\text{CORR. VEC. VELOCITY}) * K$$

where **K** is a internal constant.

The sensor value is read every 10 ms and a correction value calculated which is accumulated in a register and is used for calculation of the next interpolated step.

To obtain a more stable movement there is a ± 0.1 mm zero zone from desired location within which no correction is done.

The actual deviation from the nominal path is saved when the program is stopped (no matter how). It is in effect when the program is restarted (even at backwards execution). The correction is cleared first when a **POS**-instruction without contour argument is executed.

Limitations

The contouring function is designed for welding applications where low robot speeds are used. It is not intended for cases where large deviations from the programmed path are expected.

In general the contour function should be used at robot speeds of less than 20 mm/s and deviations of less than ± 10 degrees. The function has a time constant of around 210 ms (of which the mechanical time constant of the robot is around 120 ms).

The algorithm for calculating the position corrections is located in the robot system and therefore the function is not designed to receive pre-calculated position corrections from "smart" sensors such as seam trackers.

ABB Robotics seam tracking system - LaserTrak - operates on a different principle which means that the limitations in the speed and deviations given above do not apply.

Complementary programming notes

Normally is an analog sensor used for contouring. As can be seen from the formula given above, the scale factor and correction vector speed together determine the size of the correction.

The recommended procedure for obtaining smooth contouring is to define a small scale factor and then to adjust the correction vector speed.

A recommended scale factor value is between 0.01-0.1 and a suggested starting value for the correction vector speed is four times the robot speed.

Note that the direction on the correction vector normally should be perpendicular to the programmed robot path and that if the speed of the correction vector is less than that of the programmed points no correction to the pre-programmed path will be noticed.

The BIAS defined in the contour instruction is given as a percentage of the working range of the sensor which in turn is defined as the max-/min values provided in the list of sensor parameters. If the sensor operates from 0-10 V the min-/max values should be given as 0 and 1023 (since, an analog sensor is converted to an ten bit digital value). A BIAS of 50%, therefore, means that corrections will be calculated when the sensor value deviates from 5 V. If the sensor voltage lies outside the defined MinTMax range ADAPTrvTTY ERROR 1 will be displayed.

The correction vector operates in hand coordinates or base coordinates depending on whether the sensor is defined as being ON or OFF the robot in the sensor parameter definition. "ON the robot" means that the correction vector follows the orientation of the robot hand. "OFF the robot" means that the correction vector is fixed relative the robot's base coordinates.

Note!

A correction vector cannot be defined as a pure rotation: a correction vector must be defined between two separate TCP points.

Speed

Since the contouring function is intended to be used when small deviations from the programmed path are expected, the robot speed with contouring is approx. the same as the programmed speed of the robot without contouring.

Where control of the speed of the robot is important, the VELOCITY CONTROL function should be used.

10.6 Correction vector

During free searching (direction search and counter search) the correction is given as a vector with speed and direction. When programming a correction vector, the speed is specified as a percentage of the basic speed. The direction is defined by running the robot from a start point to a final point in the direction specified for the vector.

For analog sensors, a signal level at which no correction is required is defined. (For two bit sensors, the zero level applies).

All signals lower than this level give a correction in the direction of the correction vector and all signals higher than the level give corrections in the opposite direction of the vector.

The speed value, V, in an instruction in which a correction vector is defined, has different meanings for free searching and for contour tracing as follows:

- V specifies the maximum search speed with free searching (as a percentage of the basic speed).
- V specifies a gain factor for position correction when contour tracing.

The formula for calculation of the correction, at counter tracing is given in section 10.5.

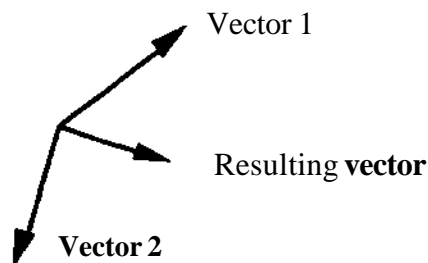
The formula for the speed of a correction during free searching is as follows:

Search speed (mm/s) = 20 x scale factor x the difference between required and actual sensor signal.

If the current sensor signal deviates less than $\pm 3\%$ from the required sensor signal no correction movement is performed (zero zone).

The max. value for the search speed is determined by the programmed speed, in mm/s.

When two or three sensors with associated correction vectors together control a free search or contour tracing, the different directions and speeds will be stored as shown in the example below.



The robot can locate an object in a plane or follow an optional contour with two correction movements. An object in space can be localized by the robot with three correction movements.

When a sensor, **in accordance with sensor data entered** (see section 9.5) is mounted on the robot, the associated correction movement follows the robot orientation. If the robot wrist is rotated around the TCP, the direction is affected. When a sensor, according to sensor data entered, is installed elsewhere, the above does not apply.

A correction movement for a specific sensor can be programmed in a program at any time. It applies until:

- A new movement for the same sensor is programmed and activated.
- The contents of the program memory are lost.

The VECTOR function is used when the direction and the speed of a correction movement are to be determined.

When the instruction is executed, the new correction vector will replace any previous vector for the sensor concerned.

The following applies for a correction vector:

- A correction vector for a sensor is programmed. If several vectors for several sensors are to be program, one instruction is required for each sensor.
- The correction vector for a sensor applies until a new movement for one and the same sensor is programmed and activated
- A program correction vector follows the orientation of the robot if the sensor concerned is mounted on the robot in accordance with the sensor data read in.
- If the memory content in the program memory is lost, all vectors disappear
- At start-up, no correction vectors are programmed.
- At Testart the correction vectors remain active.

10.7 FRAME

The procedure for programming automatic definition of FRAME is described in detail in the following pages. Note that the method for searching for objects must always be adapted from case to case. The basic definition of the FRAME function is found under section 3.9.2.

A simple example of the appearance of a program section with automatic definition of program displacement is given below. It is assumed that the program is to be twisted in a plane.

1. Ensure that program displacement 0 is active. Displacements stored on each other give an inaccurate result.
2. Place the object to be located in a suitable place.
3. Program the start point for searching for the actual location of position 1, (FINE position)
4. Program the end point for the searching so that the searching is always successful if an object is available, (FINE position).
5. Program an instruction with the FRAME function under the POSITIONING menu for position 1.
6. Perform points 3-5 above for position 2.
7. Perform points 3-5 above for position 3.

The program is now to have the following appearance. Note that all instruction numbers and numerical values in the instructions are examples only.

90	SET FRAME 0	Normal position for program displacement, i.e. no displacement is activated.
100	POS V=100% FINE	Start point for searching for the true location of position 1.
110	POS V=10% FINE SEARCH SI	Searching for the position 1 true location.

		10 Adaptivity	
120	POS V=10% FRAME POS 1	Definition of displacement of position 1.*	
130	POS V=100% FINE	Start point for searching for the true location of position 2.	
140	POS V=10% FINE SEARCH S2	Searching for the position 2 true location.	
150	POS V=10% FRAME POS 2	Definition of displacement of position 2.*	
160	POS V= 100% FINE	Start point for searching for the true location of position 3.	
170	POS V=10% FINE SEARCH S3	Searching for the position 3 true location.	
180	POS V=10% FRAME POS 3	Definition of displacement of position 3.*	

8. Enter instruction 190 with the help of the FRAME instruction under the LOGICAL INSTRUCTION menu. Select the sub-instruction DEF. The result is to become as follows:

190 DEFINE FRAME 1

*)Note

When these positions are programmed, the robot is to stand in the positions which correspond to the search points on a non-displaced object. Each FRAME-POS instruction has to be executed once were the non-displaced positions are. The execution of these instructions will give information for the instruction DEFINE FRAME later on were the non-displaced object were placed.

For a successful search for the true location of the positions 2 and 3, the complete program must be parallel displaced when the searching for the true location of position 1 is performed. When the definition of FRAME 1 is performed in accordance with the example, the parallel displacement is cancelled. The program displacement according to FRAME 1 is to be activated instead. This is to be continued as follows:

9. Add an instruction with number 125 to the program. This instruction is to activate a reference point. Use:

- INSERT under the EDITING menu to create the new instruction number.
- REFP under the POSITIONING menu for activation of the reference point.

10. Program deactivation of a reference point as an instruction 200. Use the REFP under the POSITIONING menu.

11. Activate the program displacement FRAME 1 with the help of instruction 210. Use SET under the FRAME function in the INSTRUCTION menu.

12. The program should now continue with a further searching which determines how the object is parallel-displaced in relation to the established program displacement. The instructions 220 - 240 are used for this in accordance with the following:

220	POS V=50% FINE	The start point is placed well within the area occupied by the object.
230	POS V=10% FINE DIST SEARCH SI	Searching transversely along the position of one long side.
240	POS V=10% REFPOINT ON	Continued program running is performed parallel displaced and adapted to the true position of the object.

The program is now to have the following appearance:

90	FRAME 0	Any program displacement is cleared.
95	POS V=0% REFPOINT OFF	REFPOINT off
100	POS V=100% FINE	The start point for searching for the true location of position 1.
110	POS V=10% FINE SEARCH SI	Height tracing downward of the true location of position 1.
120	POS V=10% FRAME POS 1	Definition of displacement of position 1.
125	POS V=0% REFPOINT ON	Parallel displacement of the program before searching for the positions 2 and 3 is activated.
130	POS V=100 %FINE	Start point for searching for the true location of position 2.
140	POSV=10% SEARCH S2	Transverse searching for the true location of position 2.
150	POS V=10% FRAME POS 2	Definition of displacement of position 2.
160	POS V=100% FINE	Start point for searching for the true location of position 3.
170	POS V=10% FINE SEARCH S3	Transverse searching for the true location of position 3.
180	POS V= 10% FRAME POS 3	Definition of displacement of position 3.
190	FRAME 1 DEFINE	Position of FRAME 1 is calculated on the basis of the three searches above.
200	POS V=0% REFPOINT OFF	Parallel displacement of the program is deactivated.
210	FRAME 1	The program displacement is activated.

220	POS V=50% FINE	<div>10 Adaptivity</div> <div>The start point is placed well within the area occupied by the object.</div>
230	POS V=10% FINE SEARCH SI	Transverse searching along the position of one long side.
240	POS V=0% REFPOINT ON	Continued program running is performed parallel displaced, adapted to the true location of the object.
400	REFPOINT OFF	
410	RETURN	

Section	Page
11.1 OVERVIEW OF FIVE PROGRAMMING MENUS	11:3
11.2 Initialization sequence for the main program	11:5
11.3 PROGRAM STRUCTURE	11:5
11.4 PATTERN PROGRAM	11:7
11.5 PROGRAMMING ROBOT PATHS	11:11
11.5.1 Smooth wrist reorientation	
11.5.2 Recommendation for Arc Welding	
11.5.3 Recommendation for Glueing and Sealing	
11.5.4 Process application type plastic cutting, metal cleaning, deburring and polishing	
11.5.5 Recommendation for Assembly, Machine, Tending and Material Handling	
11.5.6 Closely spaced positions	
11.6 PROGRAMMING EXAMPLES	11:15
11.6.1 Arc welding of rear axis member	
11.6.2 Glueing of a car door	
11.6.3 Cutting and deburring of plastic, forming of holes using the CIRCLE function	
11.6.4 Cutting and deburring of plastic lining for a car door	
11.6.5 Material handling applications	
11.7 HIGH PERFORMANCE PROGRAMMING	11:22
11.7.1 General	
11.7.2 Recommendations for best performance	
11.7.3 Movement principle choice	
11.7.3.1 Coordinate system	
11.7.3.2 Movement principle	
11.8 LOAD, PROGRAMMABLE	11:26
11.8.1 General function	
11.8.2 Wrist operations (A and P).	
11.8.2.1 Acceleration index calculation.	
11.8.2.2 Trimming of position gain index.	
11.8.2.3 Preset values	
11.8.2.4 Measures to deal with overshoots.	
11.9 TRIM, PROGRAMMABLE	11:31

)

)

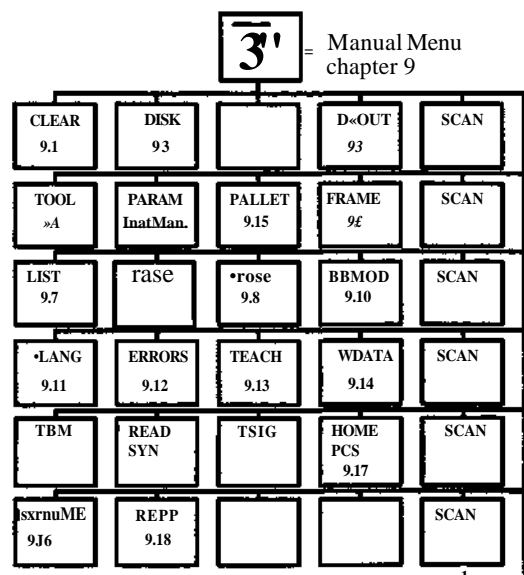
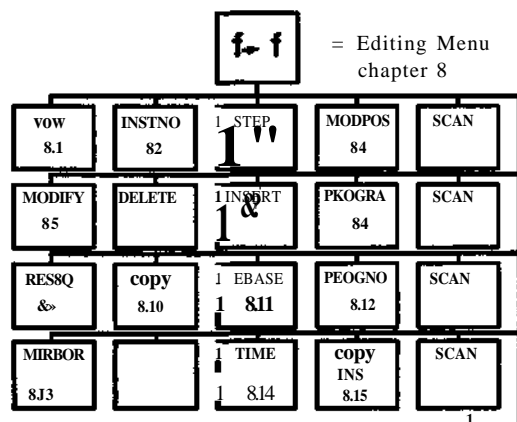
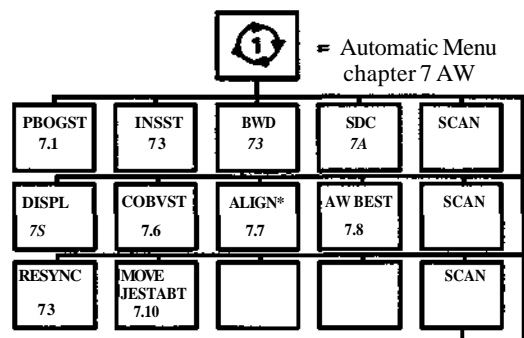
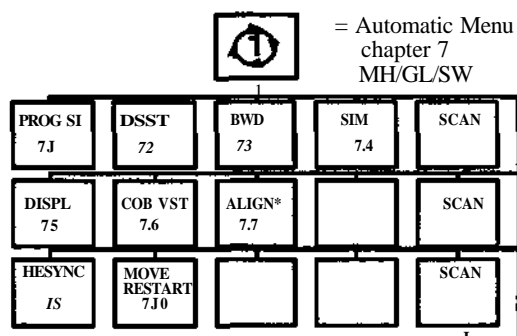
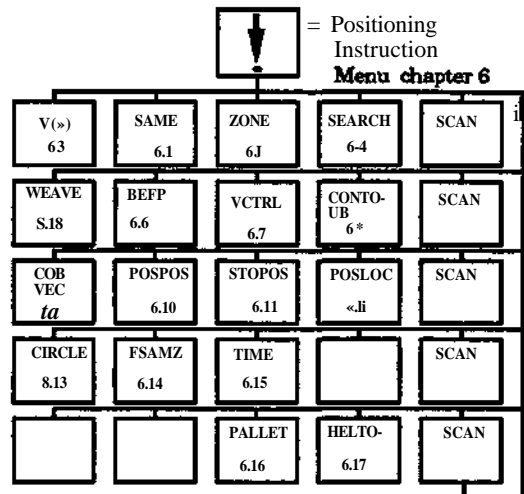
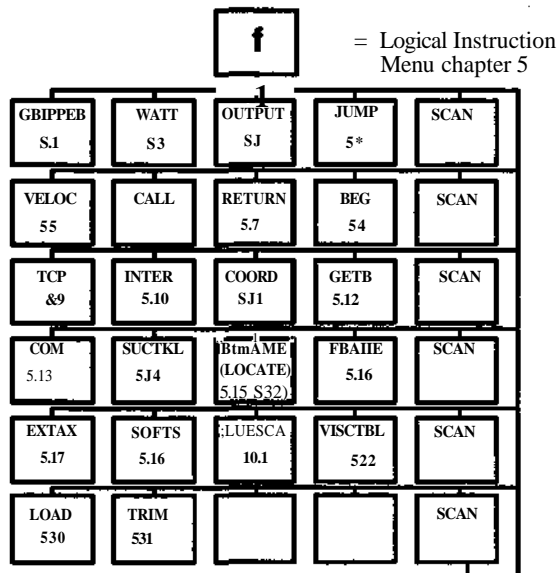
)

)

11 Programming

11.1 Overview of five programming menus:

Descriptions of the instructions are found under respective chapter and section, The section corresponds to the position on the menu. Example: TIME is described in chapter 6 section 15. etc.



11.2 Initialization sequence for the main program

It is recommended that programmers make a habit of entering the following instruction sequence with the required initialization values at the beginning of each main program when programming the robot. This ensures that programmed operations begin with correct values from the beginning.

10 V = (value) MAX = (value)	Basic and max. speeds
20 TCP (number)	Required TCP number
30 RECT COORD or ROBOT COORD	Selection of coordinate system
40 FRAME (No)	Required reference frame
50 REF P OFF	Any previously activated ref. point is removed

If these values are not specified the robot system utilizes instead the values most recently used.

The SHIFT button on the programming unit can be used to check the values currently activated program information is then presented on the display.

11.3 Program structure

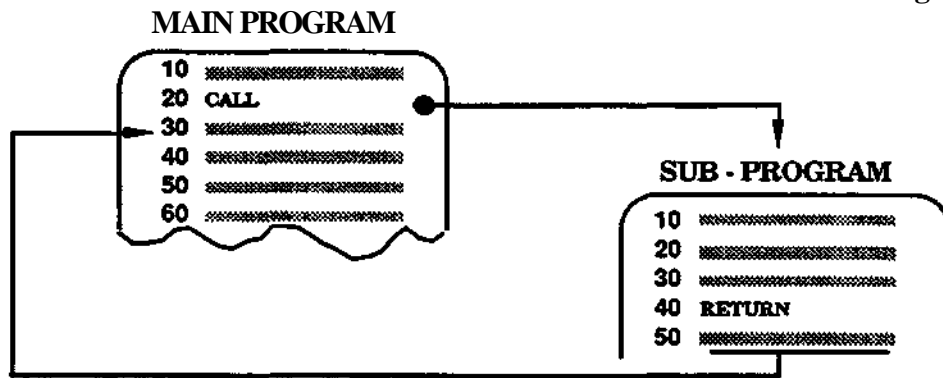
Each robot program has its own program number and consists of a series of instructions which are numbered within the program. Programs can include positioning instructions, logical instructions such as jump, allocation statements tests and comparisons, call of subprogram etc.

Programs can be numbered optionally between 0 and 9999. Instructions within a program can be numbered optionally from 1 to 65535.

The types of instructions determine how many can be accommodated in the memory at any one time. Positioning instructions occupy considerable space - more than logical instructions. The primary memory accommodates 64 kbytes which corresponds to 2500 coarse points.

An appropriate program construction includes an administrative main program which controls the work of the robot and calls subprogram for different sub-processes which are often repeated during a work cycle.

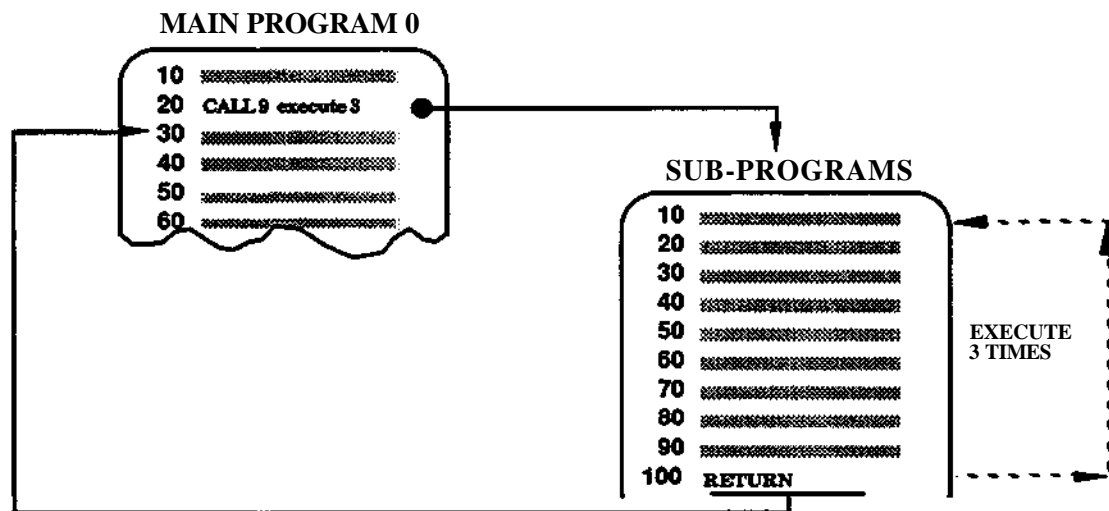
A main program with all associated subprograms is designated a program block. Program number 0 should be used for the main program if possible.



Call of a subprogram

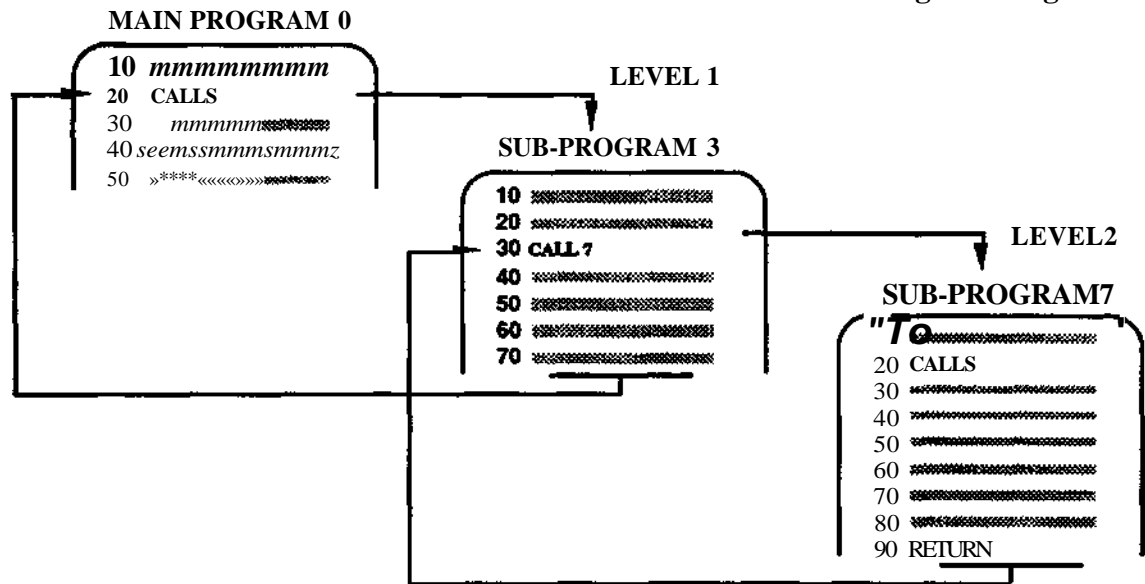
Subprogram can be called with repetition so that they are executed the number of times required before a return is made to the main program. (Max 99 repetitions)

Call of subprogram with repetition



Call of a subprogram with repetition

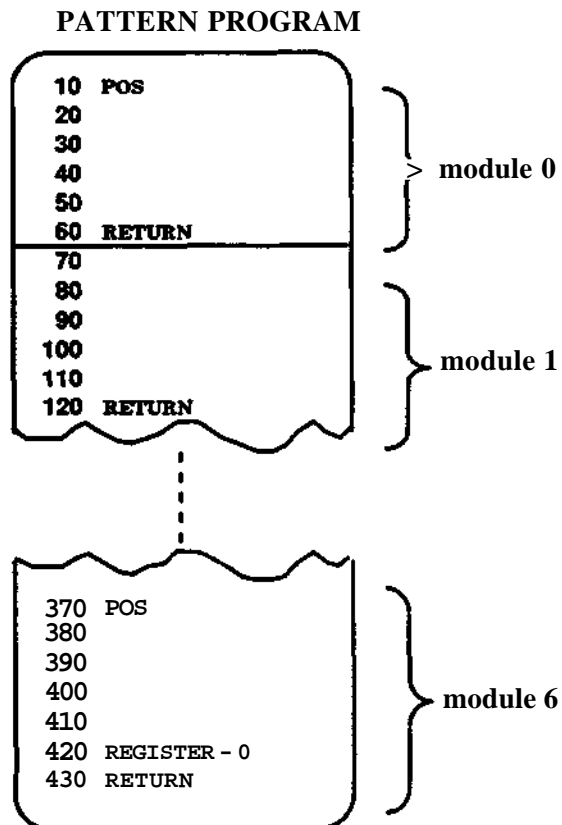
A subprogram can also call a further subprogram (maximum 12 levels down). If a subprogram on level 12 attempts to call a further subprogram, program execution is stopped and an error message is presented.



11.4 Pattern program

A pattern program is a sub-program and is treated in a special way as follows:

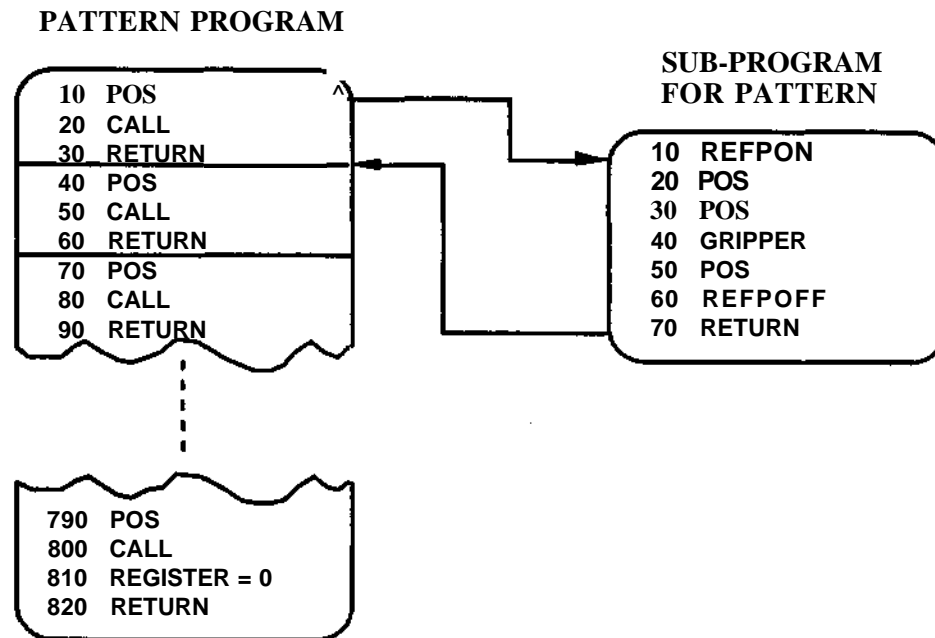
The program is divided into a number of modules separated by RETURN-instructions.



Each module can, for example, contain:

11 Programming

- A number of positions for a repetitive task where the task itself is stored within another sub-program which is called from each program module, see example below:



* Variations of a task.

- Different tasks which are executed during different stages of the main program's work cycle.

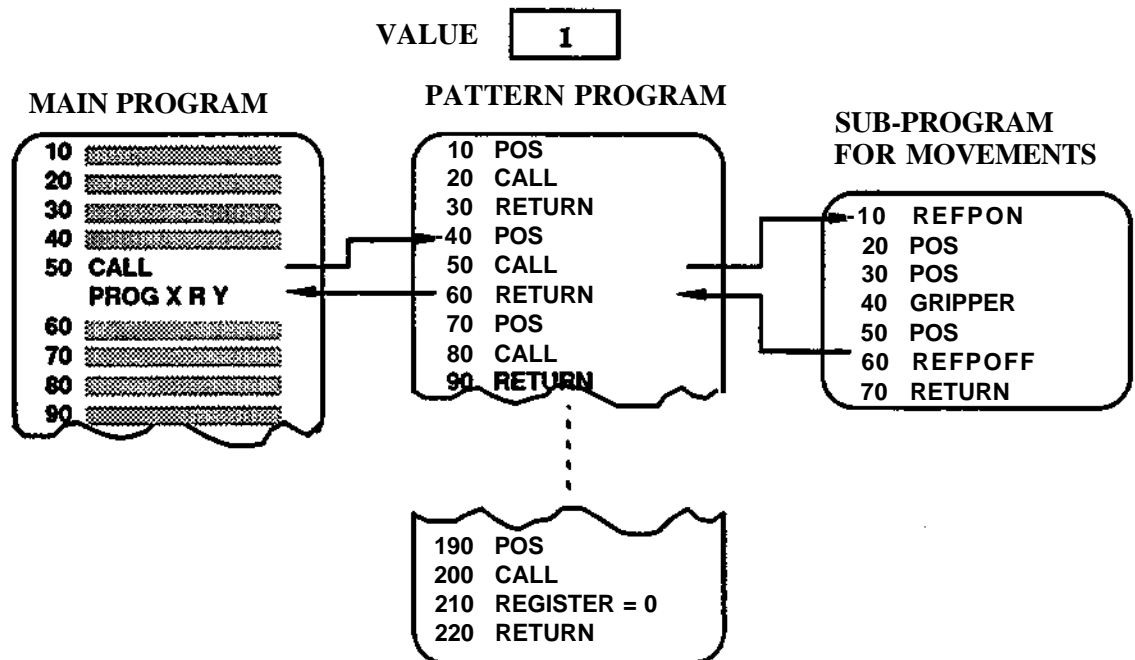
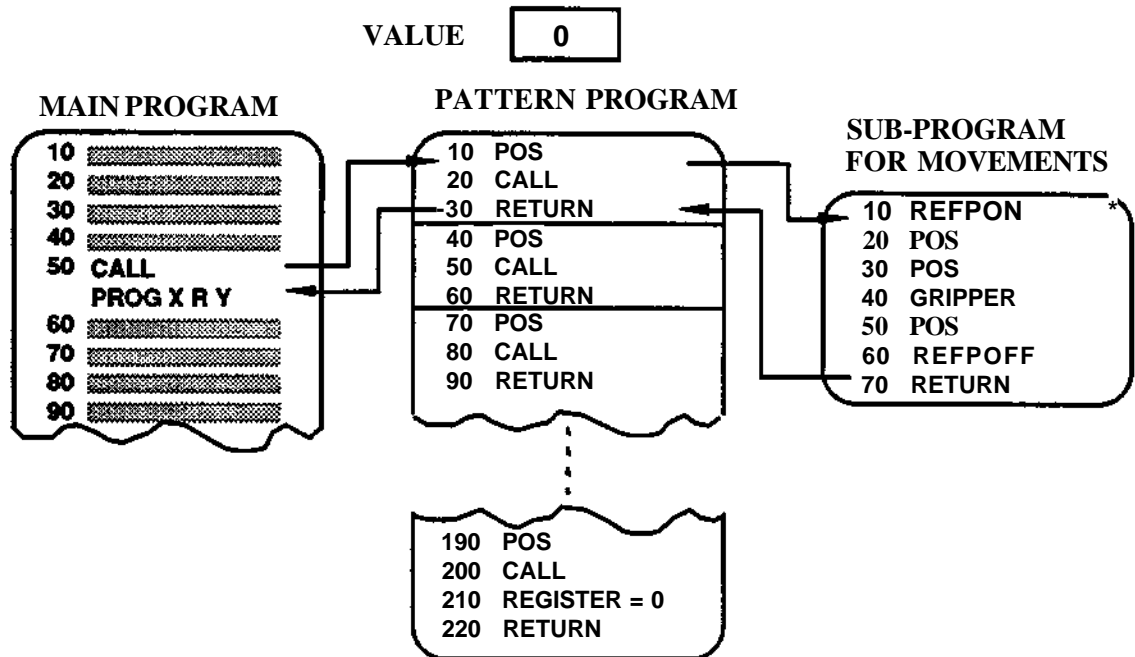
When calling such a program a number register is defined for control of which module that is to be called.

The register is normally set to 0. When the content of the register is 0, the first module is called. When the content is 1, the second module is called etc. The register is incremented with 1 each time the patternprogram is called. At the first call, the content thus becomes 1, at the second, 2 etc.

Only a register defined in such a way should be used for this purpose.

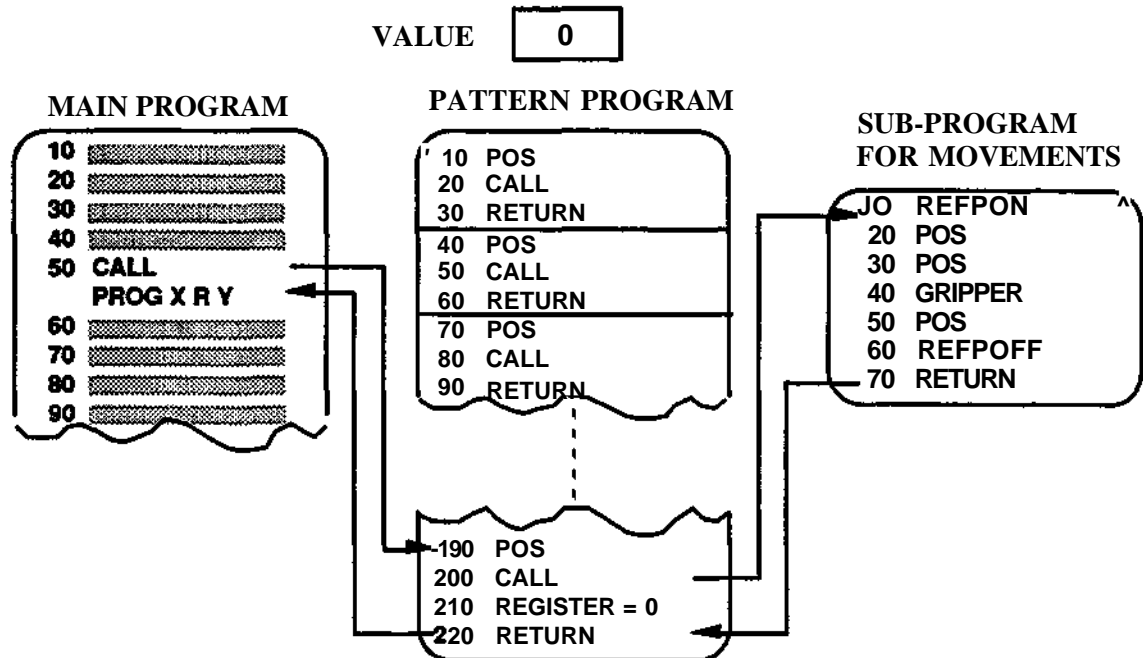
11 Programming

Below is shown an example on how the calling instruction co- operates with the number register when executing the different modules within a sub-program.



11 Programming

When programming the last module an instruction, before the RETURN-instruction, should reset the number register which controls the module execution, see figure below:



When executing a pattern program the modules can be executed in any desired order. It is not necessary to reset the register in the last module if appropriate values are inserted into the register from somewhere else within the program block.

Values can also be entered into a number register, see "Registers".

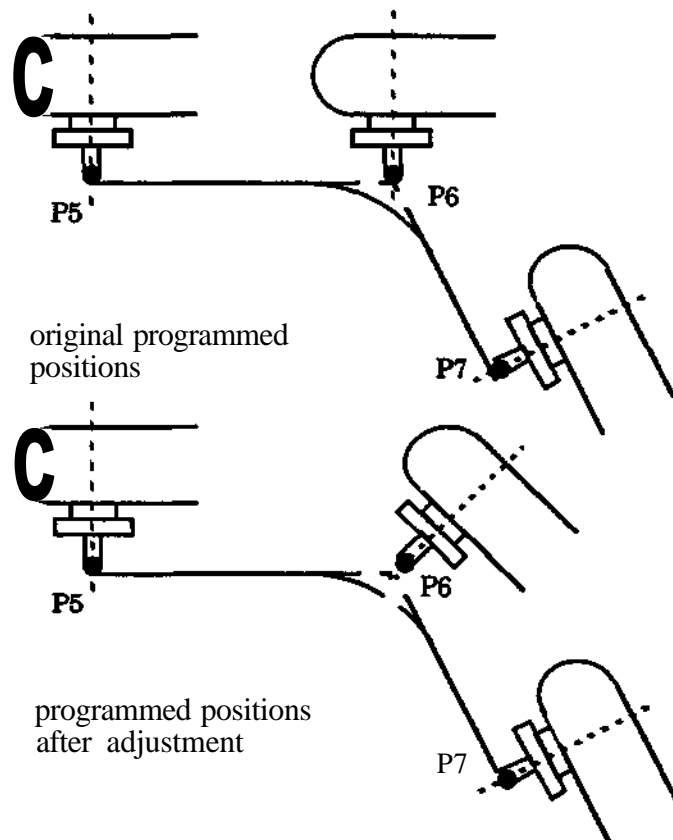
11.5 Programming robot paths

Earlier sections of the manual describe how path following and other robot instructions are affected by zone instructions (section 3.6, 6.3 etc). This section contains recommendations for programming various robot applications with respect to path following.

11.5.1 Smooth wrist reorientation

Reorientation of the wrist should be spread out over as long a path as possible to ensure fast and smooth movement. Reorientations of the wrist within a short path can limit the robot TCP velocity and lengthen the cycle time.

The example below is a case where the TCP positions P5, P6, P7 are programmed, and the reorientation of the wrist between position P6 and P7 has resulted in undesirable reduction in velocity. The reorientation will now be reprogrammed to take place between P5 and P7. (see figure)



To obtain a smooth reorientation between positions P5 and P7 the following method can be used: Depending on the profile of the path, either remove P6 temporarily or change P6 to a circle point. Run the program slowly from P5 towards P7 in rectangular coordinates. When the TCP passes P6 stop the robot. Correct the X, Y, Z location of the TCP to position P6 without changing the orientation. Program position 6 and change back from a circle point to a position.

The wrist reorientation is now spread out smoothly from P5 to P7. It is possible to spread out the reorientation even more by means of more positions, in order to achieve a smooth reorientation over a longer path.

At some TCP velocities a circular arc can give best smoothness for reorientation. This assumes that the position zones do not overlap, and that the circle radius is more than twice the zone size (to avoid overlap at entry and exit from the circle).

11.5.2 Recommendation for Arc Welding

Recommended system parameter values

FINE	2 mm	(=default)
CORNER1	5 mm	
CORNER2	-100 mm	(=default)
PATH	-25 mm	(=default)

The value 5 mm for CORNER1 is chosen to give a small corner radius while still allowing sufficient path length for smooth reorientation.

Position Zones

FINE	Used for weld start and finish positions
CORNER1	Used during the weld process. Note that the robot starts the corner path about 5 mm before the programmed position. Logical instructions and for example, change of weld data, are executed just before entry to the zone.
CORNER2	Used for fast movements between weld paths.
PATH	Used during the weld process. Note that because of the low velocity the zone is very small. The robot TCP will therefore enter the position and logical instructions will execute at the programmed position.

Program structure

Rectangular coordinates should be used during the weld process.

Robot coordinates are used for movement between process runs as this gives the highest velocity.

FINE	positions are used at weld start and finish.
CIRCLE	is used when possible on curved weld runs. Reorientation is spread out over as long a path as possible.
CORNER 1	is used on sharp turns on a weld run. Note that the distance between programmed positions should be over 10 mm to avoid velocity reduction due to zone interference.

11.5.3 Recommendation for Glueing and Sealing

Recommended system parameter values

FINE	2 mm	(=default)
CORNER1	15 mm	(=default)
CORNER2	-100 mm	(=default)
PATH	-25 mm	(=default)

Position Zones

PATH Used in glue/seal process for start of flow, change in flow etc. PATH is the only argument to the GLUE instruction. The size of the PATH zone at a TCP velocity of 150 mm/sec is 3,75 mm and at 500 mm/sec is 12,5 mm.

CORNER 1 Used in the glue/seal process.

CORNER2 Used for fast movement to and from the process.

Program Structure

Rectangular coordinates are used in the process.

Robot coordinates together with CORNER2 are used for movements to and from the process. This gives the fastest possible movement.

Circle is used for curved glueing paths.

Reorientation of the wrist is spread out over the greatest available space to give smooth reorientation. The distance between positions should be greater than two times the zone size to avoid velocity reduction at interfering zones.

11.5.4 Process application type plastic cutting, metal cleaning, deburring and polishing

Recommended parameter values

FINE	2 mm	(=default)
CORNER1	15 mm	(=default)
CORNER2	-100 mm	(=default)
PATH	-25 mm	(=default)

Use of ZONES

FINE Use at the start and end of a program as "Start" and "end" positions. Avoid in the process program because of loss of cycle time.

CORNER1 Used during the process for reorientation with large angles.

CORNER2 Used with robot coordinates for movements to and from workpiece for example between deburring operations.

PATH Used to form a path to follow the curve of a workpiece.

PATH and **CIRCLE** are very useful in corners with a circular radius, or when working on round holes. At the working speeds of 80-200 mm/sec used for cutting and deburring a good zone is obtained with **PATH**. Example: at 150 mm/sec the **PATH** zone is 3,8 mm.

Program structure

Movements in space, for example between two process machines are programmed using

- **ROBOT COORD** and **CORNER2** for fastest movements.

In a process program, for example a cutting process, straights or uneven curves

- **RECT COORD** and **PATH** in circular paths or even curves
- **RECT COORD** and **PATH** and **CIRCLE**

Workpiece reorientation is done over as long a distance as possible, or, if this is not possible use a **TIME** instruction to allow smooth reorientation.

FINE positions should be avoided in the process as the TCP pause can cause an uneven finish on the work piece.

It is important to avoid overlapping zones to obtain as even a process speed as possible. That is to say, the distance between positions must be at least double the relevant zone size.

Do not change from **ROB COORD** to **RECT COORD** during the process because of the resulting pause in TCP movement.

11.5.5 Recommendation for Assembly, Machine Tending and Material Handling

Recommended system parameter values

FINE	2 mm	(=default)
CORNER1	15 mm	(=default)
CORNER2	-100 mm	(=default)
PATH	-25 mm	(=default)

Position Zones

FINE Used in connection with fetching or hand-over of work components, I/O-handling, activation of reference points, store positions etc.

CORNER1 Used in connection with change of direction of a robot TCP in a small space, deceleration of robot velocity before a fetch or hand-over position and such cases.

PATH Used for TCP movements in movement sub-programs for the first and last positions in fetch and assembly sub-programs etc.

Program structure

The main program is used for movement of the robot, blocks of logical instructions and calls to subprograms. This program section should not require very accurate path following. The accuracy of the path can be allowed to vary with the robot TCP velocity.

Generally robot coordinates are used for this program section and PATH is used for movement between different work-centers. If positions are at a distance from each other a faster result can be obtained using CORNER2, but if positions lie close to each other PATH gives higher velocity. Where possible, logical instructions should be collected in blocks in connection with calling a sub-program or with a return from a sub-program.

Sub-programs are used where there is a requirement for accurate path following at all speeds. Rectangular coordinates are used. The choice of zone depends on the required path accuracy.

11.5.6 Closely spaced positions

To avoid problems with closely spaced positions, the distance between two positions is to be two steps at the speed currently programmed, e.g. 10 mm at 100mm/s, 50 mm at 500 mm/s.

In applications with critical cycle times and closely spaced positions, programming without call of sub-programs is recommended to minimize the number of logic instructions.

11.6 Programming examples

General

A number of functions often used when examples are to be programmed are summarized below.

A subprogram is called up as follows:

1. Press EDITING.
2. Press the function button SCAN.
3. Press the function button PROG.
4. Enter the number of the required subprogram with the numerical keyboard.
5. Press the function button ENTER.

Searching of an instruction within the program is executed as follows:

1. Press EDITING.
2. Press the function button STEP.

Now three subfunctions are displayed, which can be used, namely:

- FORW, step forward to next instruction number.
- BWD, step backwards to the subsequent instruction number.
- INST NO, jump directly to an instruction number, which is entered with the numerical keyboard.

Functions for program execution are located under the AUTO menu, see chapter 7.

Continuous program execution is commanded with the function button START P.

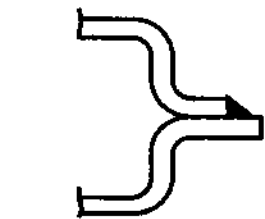
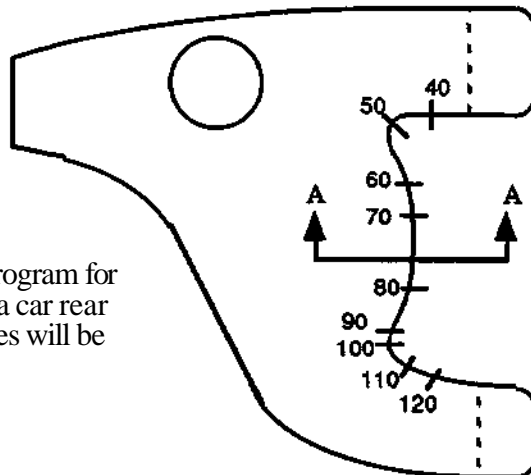
Program execution **forward**, instruction by instruction, is commanded with the function button START I.

Positioning backward, instruction by instruction, is commanded with the function button BWD.

"Satisfied conditions" for WAIT instructions are simulated with the function button SIM.

11.6.1 Arc welding of rear axis member

The example shows a program for arc welding one side of a car rear axis member. Three sides will be welded.



section A-A

Main program 0

```
10 V=2500 mm/s. Max 2500 mm/s
20 TCP = 1
30 ROB COORD
40 FRAME 0
50 POS V = 100% C2
60 WAIT until INPUT 5=1
70 CALL PROG 100
80 RETURN
```

(home position)
(wait for start signal)
(call weld program)

Weld program 100

```
10 POS V = 100% C2
20 POS V = 100% C2
30 RECT COORD
40 POS V= 10% FINE AWELD 1100
50 POS V= 10% CIRCLE
60 POS V= 10% PATH
70 POS V= 10% PATH
80 POS V= 10% CIRCLE
90 POS V = 10% PATH
100 POS V = 10% C1
120 POS V = 10% FINE WEND 1100
130 POS V = 50% PATH CALL PROG 300
140 POS V = 50% PATH
etc.
```

(move to work piece)
(move to work piece)
(start first weld)
(circle point)
(circle point)
(small radius and reorientation, use C1)
(end of first weld)
(cleaning program)

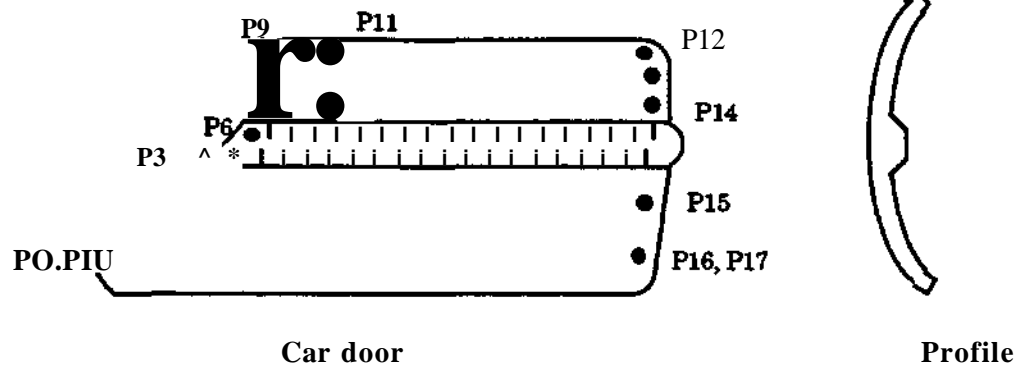
Cleaning program 300

```
10 Set output 4
20 Wait 0,5 s
30 Reset output 4
40 Wait 0,2 s
50 Set output 4
60 Wait 0,5 s
70 Reset output 4
80 Return
```

(first cleaning)

(second cleaning)

11.6.2 Glueing of a car door

**Main Program 0**

```

10 V = 2500 mm/s Max 2500 mm/s
20 TCP1
30 ROB COORD
40 FRAME 0
50 POS V = 100% C2 (home position)
60 WAIT until INPUTX=1 (wait for start signal)
70 POS V = 100% C2 (positions close to the start pos)
80 POS V = 100% C1
90 CALL Prog 1
100 RETURN

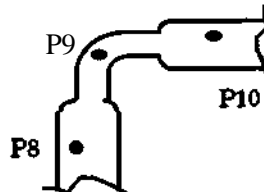
```

Glue: Process program 1

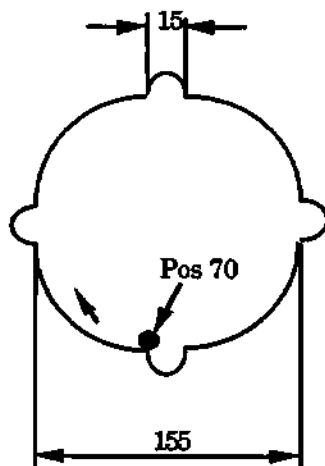
```

10 V = 500 mm/s Max 1000 mm/s
20 Pos V = 100% PATH
30 RECT COORD
40 POS V = 100% FINE (P0, above start pos)
50 POS V = 100% GLUE FLOW = 100% (PI, glue start position)
60 POS V = 100% PATH (P2)
70 POS V = 100% CIRCLE (P3)
80 POS V = 100% PATH (P4)
90 POS V = 100% PATH (P5)
100 POS V = 100% CIRCLE (P6)
110 POS V = 100% PATH (P7)
120 POS V = 100% PATH (P8)
130 POS V = 100% C1 (P9, note 1)
140 POS V = 60% GLUE FLOW = 30% (P10 reduced flow)
150 POS V = 100% GLUE FLOW = 100% (P11 increased flow)
160 POS V = 100% C1 (P12)
170 POS V = 60% GLUE FLOW = 30% (P13 reduced flow)
180 POS V = 100% GLUE FLOW = 100% (P14 increased flow)
190 POS V = 100% PATH (P15)
200 POS V = 100% PATH (P16)
210 POS V = 100% GLUE FLOW = 0% (P17, glue end position above P16)

```



11.6.3 Cutting and deburring of plastic, forming of holes using the CIRCLE function



In this example it is not necessary to reorient the work-tool.

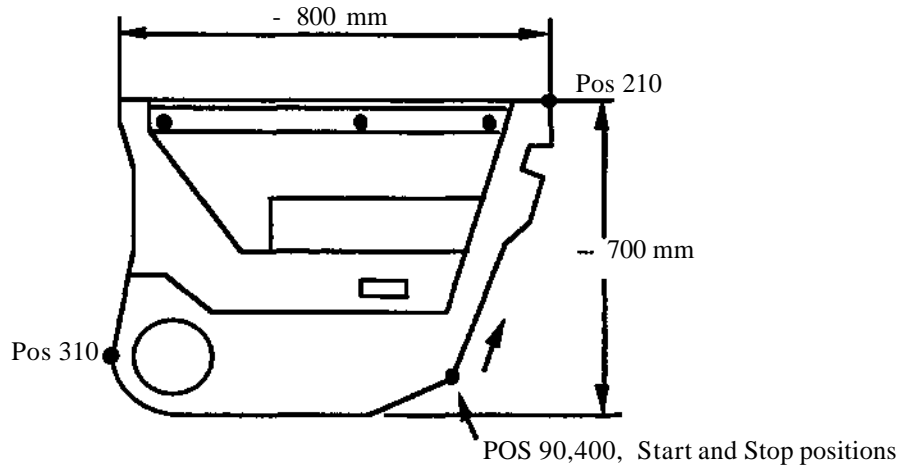
Program 10

```

10  V = 1000 mm/s max 2500 mm/s
20  TCP=1
30  RECT COORD
40  PosV= 100% PATH (start position)
50  WAIT till INPUT 10=1
60  Pos V = 50% PATH (in front of work piece)
70  Pos V = 5% PATH (movement to work piece)
80  Pos V = 10% CIRCLE (cutting large circle)
90  Pos V = 5% PATH
100 Pos V = 5% CIRCLE (cutting small circle)
110 Pos V = 5% PATH
120 Pos V = 10% CIRCLE
130 Pos V = 5% PATH
140 Pos V = 5% CIRCLE
.
.
230 Pos V = 5% PATH (as pos 70, cutting finished)
240 Pos V = 10% PATH (movement from work piece)
250 Pos V = 100% PATH (return to start position)
260 WAIT till INPUT 20=1

```


11.6.4 Cutting and deburring of plastic lining for a car door



In this example the work-tool is reoriented.

Program 100

```

10 V = 1000 mm/s max 2500 mm/s
20 TCP = 1
30 Robot Coord
40 Pos V= 100% PATH (to start position)
50 WAIT till INPUT 1 = 1
60 Pos V = 100% PATH (in front of work piece)
70 Rect Coord
80 Pos V = 50% PATH (movement to workpiece)
90 Pos V = 5% PATH (tool to work piece)
100 Pos V = 5% PATH (cutting starts)
.
.
180 Robot Coord
190 Pos V= 100% PATH (reorientation of workpiece in free space)
200 RECT COORD
210 Pos V = 50% PATH (movement to work piece)
220 Pos V = 5% PATH (tool to work piece)
.
.
310 PosV = 5% C2 (cutting in a circular shape)
320 Pos V = 5% CIRCLE
330 Pos V = 5% C2
.
.
400 Pos V = 5% PATH (cutting finished)
410 Robot Coord
420 Pos V = 100% PATH (tool from work piece)
430 Pos V = 100% PATH (to start position)
440 WAIT until INPUT 2 = 1

```


11.6.5 Material handling applications

When logical instructions must be executed at the programmed position a FINE zone is used. Define and activate a suitable TCP in the sub-program. This will help to ensure that the robot will move to the correct programmed position during program testing, editing etc. Define and activate a suitable TCP and Basepoint for assembly applications which include twisting around a fixed center point. This simplifies programming. When positions with displacement in the X, Y, Z axes relative to a reference point are used, a FINE zone is programmed where the reference point is activated. When positions with displacement in the X, Y, Z-axes relative to a stored position are used, a FINE zone is programmed where the position is to red (STO POS). A program layout can be as follows:

Program 10

```

10  TCPO
20  ROBOT COORD
30  POS V=100% PATH
40  POS V= 100% PATH
50  POS V= 100% PATH
60  CALL PROG 110      40    POS
70  TCPO
80  ROBOT COORD
90  OUTPUT 10=0
100 REG1=REG1 + 1
110 POS V= 100% PATH
120 POS V= 100% PATH
130 POS V=100% PATH
140 POS V=100% PATH
150 WAIT TILL IN 8=1
160 OUTPUT 11=1
170 CALL PROG 120
180 TCP 0
190 ROBOT COORD
etc

```

Program 110

```

10  TCP 1
20  RECT COORD
30  GRIPPER 1 OPEN WAIT =,5 S
V=    100%    PATH
50  POSV=100%C1
60  POSV=100%C1
70  POS V=5% FINE
80  GRIPPER 1 CLOSE WAIT 0,3 S
90  POSV=10%C1
100 POS V=25% PATH
110 RETURN

```

When first programming these applications the robot may enter a fault condition. If and when this may happen depends on possibilities which are then shown on the display. To handle these situations it is important that the programmer structures the program in an operator-friendly way.

Therefore:

1. Follow the recommended program structure.
2. After a return to the main or transport program from a sub- program use the following three- instruction sequence:

```
TCPX
ROBOT COORD
POS V=100% PATH
```

The position should lie about 200 mm above or in front of the fetch or hand-over position. This position results in controlled path following even if the operator by mistake makes a restart from the end of the sub-program. Instruction TCP X is also a safety measure if something has happend in the sub-program. ROBOT COORD gives the fastest cycle times.

3. The first position in a sub-program should be the same as the last position in the main program. This will result in a controlled path following even if the operator by mistake restarts from the sub-programs first instruction.

These recommendations will allow the operator to proceed with the operations after a fault condition without needing to restart from the beginning of the program, and without needing to know exactly at which robot position the stop occured. The operator can read-in the first or last instruction in the sub-program depending on whether it is desired to repeat or hop over the operation which failed.

11.7 HIGH PERFORMANCE PROGRAMMING

11.7.1 General

Two servo concepts (mode A and B) ensure high performance of IRB 6000. In mode A advanced algorithms are used to handle flexible mechanical arm structures. Thereby, it will be possible to allow very rapid motor torque changes, resulting in very fast movements.

One of the two servo modes is automatically chosen:

- Mode A for short movements for axis 1, < 120 mm, with high programmed speed.
- Mode B is for longer movements or short movements at low speed.

The difference in cycle time between mode A and mode B is especially large for short axis 1 motions.

The maximum TCP motion distance for mode A varies between approx. 60 mm and 120 mm, depending on the TCP position in the workspace. Mode A will not be activated for the main axes if any of the wrist axes performs a too long motion.

Mode A is never activated during motions within corner zone (path, corner 1 or corner 2) or immediately before or after the corner zone.

11.7.2 Recommendation for best performance

Use robot coordinates the whole time, which results in smooth and fast movements.

At spot weld programming it is suitable to set the parameter ZONE to PATH=-25, C1=15 and C2=-100. (Minus before the number means that this zero zone is dependent of the velocity, directly proportional against the programmed velocity.)

Value -25 means one computer step through the zero zone irrespective of the programmed velocity (see fig. 1).

Value 15 means a fixed zero zone irrespective of the programmed velocity, suitable when one wants to program high velocity without increasing zero zones. This results in a dynamical velocity reduction through the zero zone.

Value -100 means 4 or more steps through the zero zone, which reduces the speed (rarely used in spot weld programs).

This means that the smallest velocity reduction is obtained when PATH= (-25) is used. **For** each extra computer step inside the zero zone the cycle time is increased by 0.05 sec.

Overlapping zero zones also results in speed reduction. The speed reduction increases more than described above, if the overlapping is larger, see fig. 2.

This is not valid when zero zone -25 is used.

In this case big path faults can appear, if the zero zones are overlapping, see fig. 3.

Fig.1

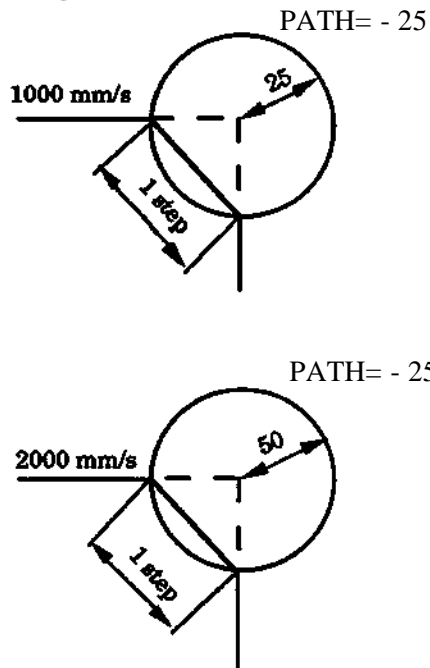


Fig.2

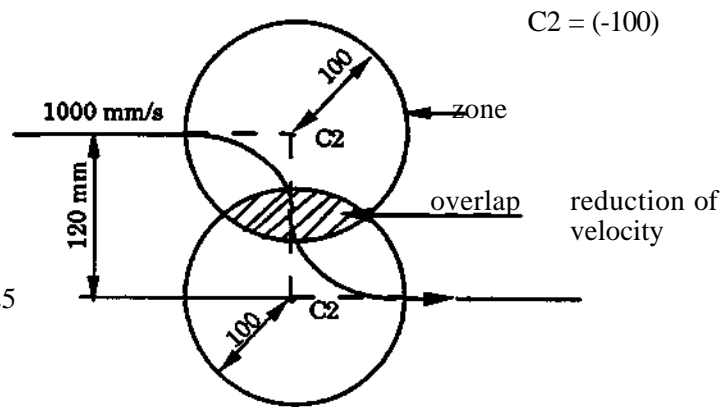
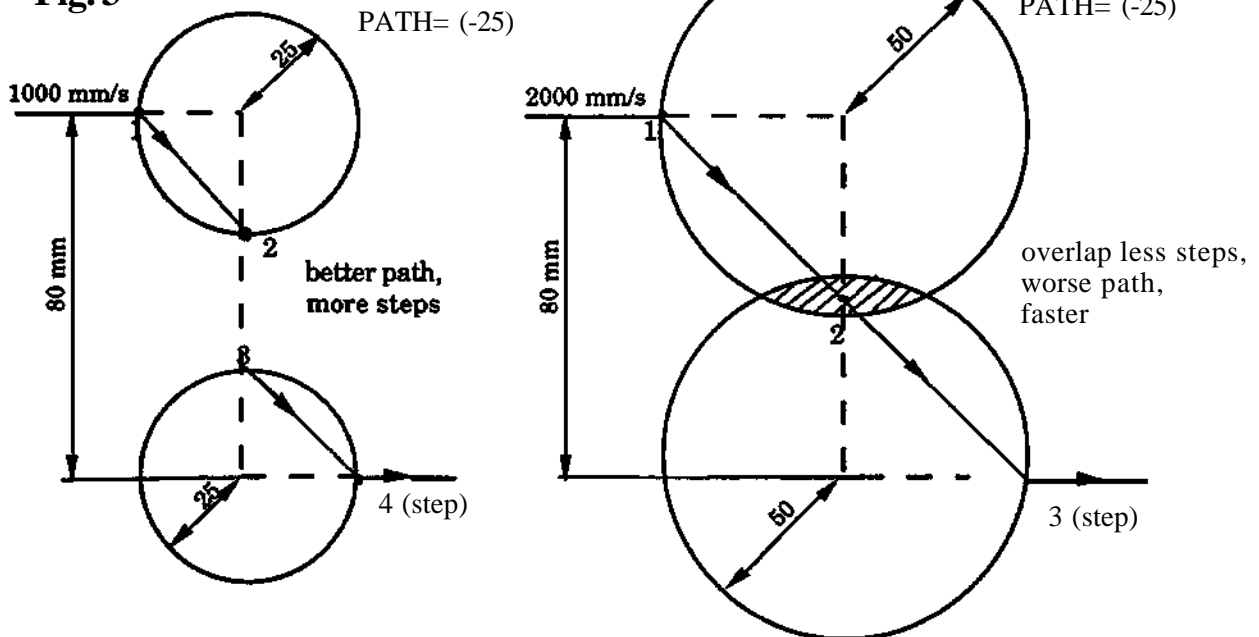
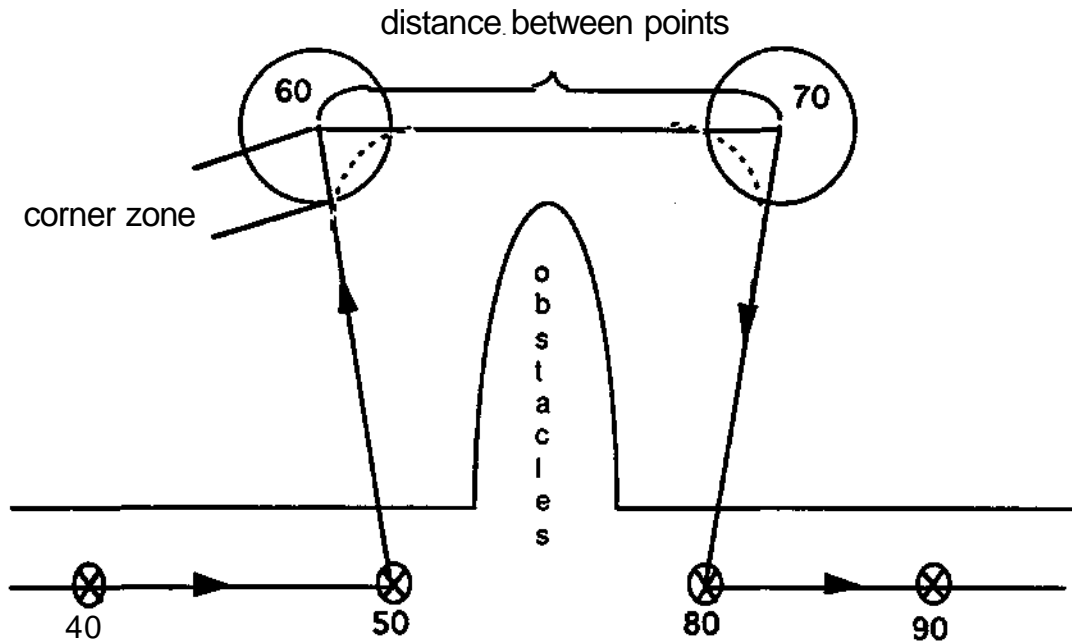


Fig.3



Above mentioned recommendations can be summarized according to the rules of thumb on the next page:

Example around obstacles:



Programming example, distance between points = 80 mm:

```

10    V=1000 mm/s Max= 2500 mm/s
20    ROBOT COORD
.
40    POS V=250 % FINE W(WELD)
50    POS V=250 % FINE W(WELD)
60    POS V=100% PATH
70    POS V=100 % PATH
80    POS V=100% FINE WCWELD)
90    POS V=250 % FINE W(WELD)
.
.
.

```

Rules of thumb

distance between points (mm)	type of corner (parameter)	velocity (mm/s)
20-50	path (-25)	500
50-125	path (-25)	1000
>125	path (-50)	2500

11.7.3 Movement principle choice

11.7.3.1 Coordinate system

When very short cycle time is required as e.g. in spot welding, ROBOT coordinates should be used. In applications where path following is more important than short cycle time, RECT-coordinates is the best choice.

Using RECT-coordinates, servo path following (SPF) is carried out, see section 3.6.1.

11.7.3.2 Movement principle

Principle 0: (param. AUTO/PATH=)

Coner paths are generated according to section 4.2.

When changing from ROBOT - to RECT-coordinates, the movement is stopped.

Principle 1: (param. AUTO/PATH=)

The problem with overlapping zero zones can be avoided, because no corner paths are generated.

The movement is not stopped when changing between ROBOT- and RECT-coordinates. Worse path following in RECT-coordinates.

11.8 LOAD, PROGRAMMABLE

11.8.1 General function

This function is only accessible for IRB 6000 and means that the robot's load can be selected and altered manually or direct in the robot program, in the same way as TCP. This means that the robot can operate optimally even when great weight differences are involved in the same program. This avoids cycle time losses.

The compensation is performed for the wrist load. Any equipment mounted on robot arm may also be defined as an additional load. This load is part of the parameters and cannot be altered during program execution.

Wrist load definition involves the following parameters:

- Wrist weight.
- Distance in X direction to the centre of gravity of the wrist load, in mm from mounting flange.
- Acceleration index (A4.. A6) for axis 4..6.
- Position adjustment index (P4..P6) for axis 4..6.

20 different loads can be defined under the MANUAL button.

Load no. 0 is defined under the PARAM menu. Load no. 1-19 are defined under menu TOOL (see chapter 9.5).

The relevant load is activated when a special load instruction, LOAD, is executed, or manually under menu TOOL.

The weight for any equipment on the robot arm is defined, under PARAM, (see Installation S3) in the same way as the wrist load, but cannot be activated by a program instruction.



The weight and distance values control the main axis's operations and should precisely correspond to the load concerned. Values that are too low may lead to overload in the mechanical structure and severe arm deflections. Values that are too high cause unnecessary cycle time loss and somewhat oscillative performance.

11.8.2 Wrist operations (A and P).

The wrist's preset acceleration and position adjustment correspond to a load with maximum permitted weight and a moment of inertia within the load diagram, although these may be altered by the user.

In applications where wrist loads with higher moments of inertia (J_o) than the load diagram permits, the acceleration and controller gain can be reduced by the user. For loads with low moment of inertia and a positive margin to the load diagram it is also possible to increase wrist operations to further optimize cycle time.

11.8.2.1 Acceleration index calculation.

The A value is controlled by two different calculations depending on the load's moment of inertia.

Procedure:

- Calculate K
- Read A in diagram next page.

11.8.2.1.1PT- loads

Press tending operation involves using large sheets extending out some distance and with a high moment of inertia. This may result in greater forces and torque loadings than permitted on the wrist.

At high loadings the robot reaches the current limit, which may result in oscillative behaviour and unpermitted mechanical loadings, especially in gears and structures in the wrist.

To reduce such loadings, the acceleration performance has to be reduced.

K has to be defined so as to utilise available torque without exceeding the permitted transverse force loading.

E is the smaller value out of K1 and K2.

Axis 4 and 5

Other variants	2.4-150 kg variant
$K1 = (950 - M_s) / J_{xx}$	$K1 = (1350 - M_s) / J_{xx}$
$K2 = 47 / (A_x * m^{0.5})$	$K2 = 70 / (A_x * m^{0.5})$

where:

$J_{xx} = J_{a5}, J_{b4}$ or J_{b5} according to Description 1KB 6000

M_s is the static moment on axis 4 or 5. $M_s = A_x * m * 9.81$

A_x in metres as per definition in diagram in section 11.8.2.1.2

m is the total handling weight.

Axis 6:

In the case of axis 6 there will be a distinction depending on what the robot cycle looks like but the same formulae apply to all variants.

$$K1 = 750 / J6$$

$$K2 = 47 / (A_y * m^{0.5})$$

Robot cycle according to case B in Description IRB 6000

$$K1 = (550 - M_s) / J6$$

$$K2 = 25 / (A_y * m^{0.5})$$

where:

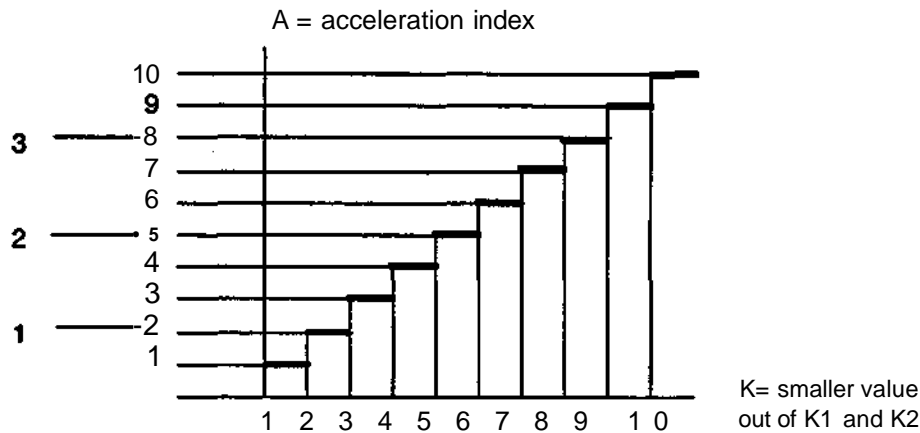
M_s is the static moment on axis 6, $M_s = A_y * m * 9.81$

J_6 is the moment of inertia on axis 6,

m is the handling weight

A_y is in meter.....see diagram in section 11.8.2.1.2

150 kg Other
 variants



11.8.2.1.2 Low loads, $J_o < 10 \text{ kgm}^2$ and weight within the load diagram

In the case of loads with moment of inertia lower than 10 kgm^2 and handling weight within the load diagram, the torque margin is regarded as utilised for increased acceleration of the wrist axis.

The following applies to all IRB 6000 variants.

A parameter K which is proportional to the margin to the load diagram's limitation for the relevant handling weight is created.

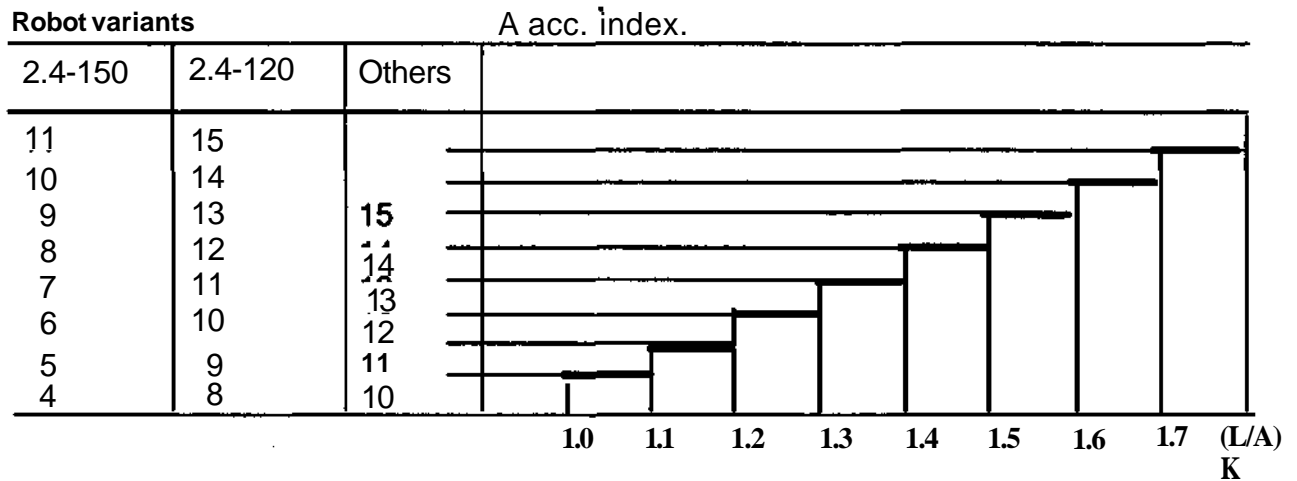
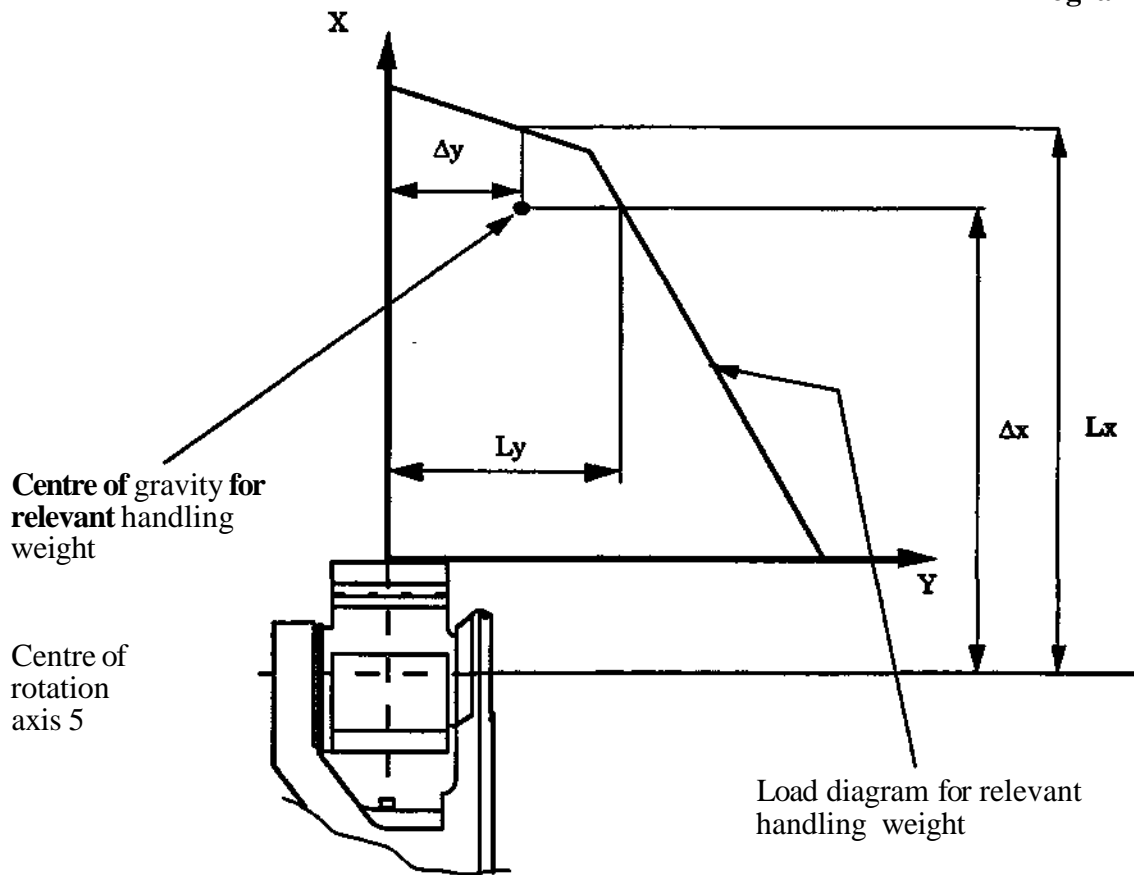
Axis 4 o5 $K = L_x / A_x$
Axis 6 $K = L_y / A_y$

L_x is the X load diagram for relevant handling weight +0.2 m for axis 4 and

L_y is the Y load diagram for the relevant handling weight for axis 6

A_x is the relevant distance of the centre of gravity for the total handling weight in the X direction from the mounting flange + 0.2 m for axis 4 and 5

A_y is the relevant distance of the centre of gravity for the total handling weight in the Y direction from the mounting flange.



Note: Because of stability limitations the actual acceleration may remain at a limited level despite the index rising.

11.8.2.2 Trimming of position gain index.

'P' denotes the index for position gain. It controls stability and settling and should be trimmed after the correct acceleration has been calculated and activated. "1" denotes very slow and stable settling, "15" denotes rapid and unstable behaviour.

Position gain depends on the load's moment of inertia and structural flexibility. When it is desired to alter the value, it should be trimmed as follows:

- Place the robot in a position corresponding to the greatest moment of inertia (in the program that is to use the load) for the axis that is to be trimmed.
- Write a program that only concerns the axis that is to be trimmed in a long movement at maximum speed.
- Alter the position gain index until a distinct performance without overshoot is achieved.

Note! SOFT SERVO activated.
As the position gain affects controller stiffness, it should be trimmed before the softness parameter is determined.

11.8.2.3 Preset values.

Robot type	Index					
	A4	A5	A6	P4	P5	P6
IRB 6000/2.4-100	10	10	11	10	11	12
IRB 6000/2.4-150	4	4	5	4	13	9
IRB 6000/2.8-100	10	10	11	10	11	12
IRB 6000/3.0-75	10	10	11	10	11	12
IRB 6000/S3.0-100	10	10	11	10	11	12
IRB 6000 £.4-120	8	8	8	8	8	8
IRB 6000 £.25 PE-75	10	10	11	10	11	12

The acceleration and position gain are set automatically to these predefined values when the weight and offset are undefined (LOAD 1..19) even if they are altered for the installation load (LOAD 0).

11.8.2.4 Measures to deal with overshoots.

Wrist overshoot may occur during trimming work if the acceleration and/or the position gain are too high.

Too high acceleration value: the motor torque reaches its maximum limit and overshoot occurs because there is not sufficient torque to stop the movement.

Too high position gain: the controller becomes too quick for the load's inertia and structural flexibility, thereby leading to oscillations.

11.9 TRIM, PROGRAMMABLE

See section 5.21.

12 Arc welding

Section	Page
12.1 Introduction	12:3
12.2 System principles	12:3
12.3 Program structure	12:12
12.4 Description and programming of welding data	12:15
12.5 Description of robot instructions	12:61
12.5.1 Instructions for the welding process	
12.5.2 Instructions for manipulator	
12.6 Programming of robot instructions	12:64
12.6.1 Define a position with a PATH zone	
12.6.2 Instructions for the weld process	
12.6.3 Instructions for positioner movements	
12.6.4 Common functions	
12.6.5 EXTFRAME	
12.6.5.1 Alignment of External axes	
12.7 Program execution	12:83
12.7.1 Program test	
12.7.2 Override function	
12.7.3 Supervision of welding process	
12.7.4 Execution of welding instructions	
12.8 Program example	12:88

)

)

)

)

12 Arc welding

12.1 Introduction

The contents in this chapter are only valid for arc welding robots.

This chapter describes operation of the robot for arc welding and the special function developed for this application.

The design of a robot welding station is described first and then how the robot interacts with the welding equipment and other peripheral equipment. The general principles and methods of preparing a program for robotized arc welding are discussed.

After this broad description of the arc welding system and program structure, detailed instructions for how the welding data (parameter values for the welding process) is programmed and used and how the robot instructions specific to arc welding applications are given.

A special section contains a description of program execution, how a program is tested and optimized, how the override function is used and how the welding process is monitored. In conclusion, an example of a robot program for welding of a simple workpiece is given.

12.2

System principles

Example of station design

A robot station for arc welding can consist of the following main parts:

- Robot
 - manipulator
 - control system
- Welding equipment
 - current source
 - wire feed
 - welding gun with hose package
 - spatter cleaning equipment
- Positioning equipment
 - mechanical unit
 - drive equipment
 - fixtur
- Sundries
 - operators's panel
 - adaption unit
 - smoke extractor
 - cooling unit
 - safety equipment

12 Arc welding

The requirement for exchange of signals between the robot and the other equipment depends on the configuration concerned but the basic requirement is in accordance with the following list:

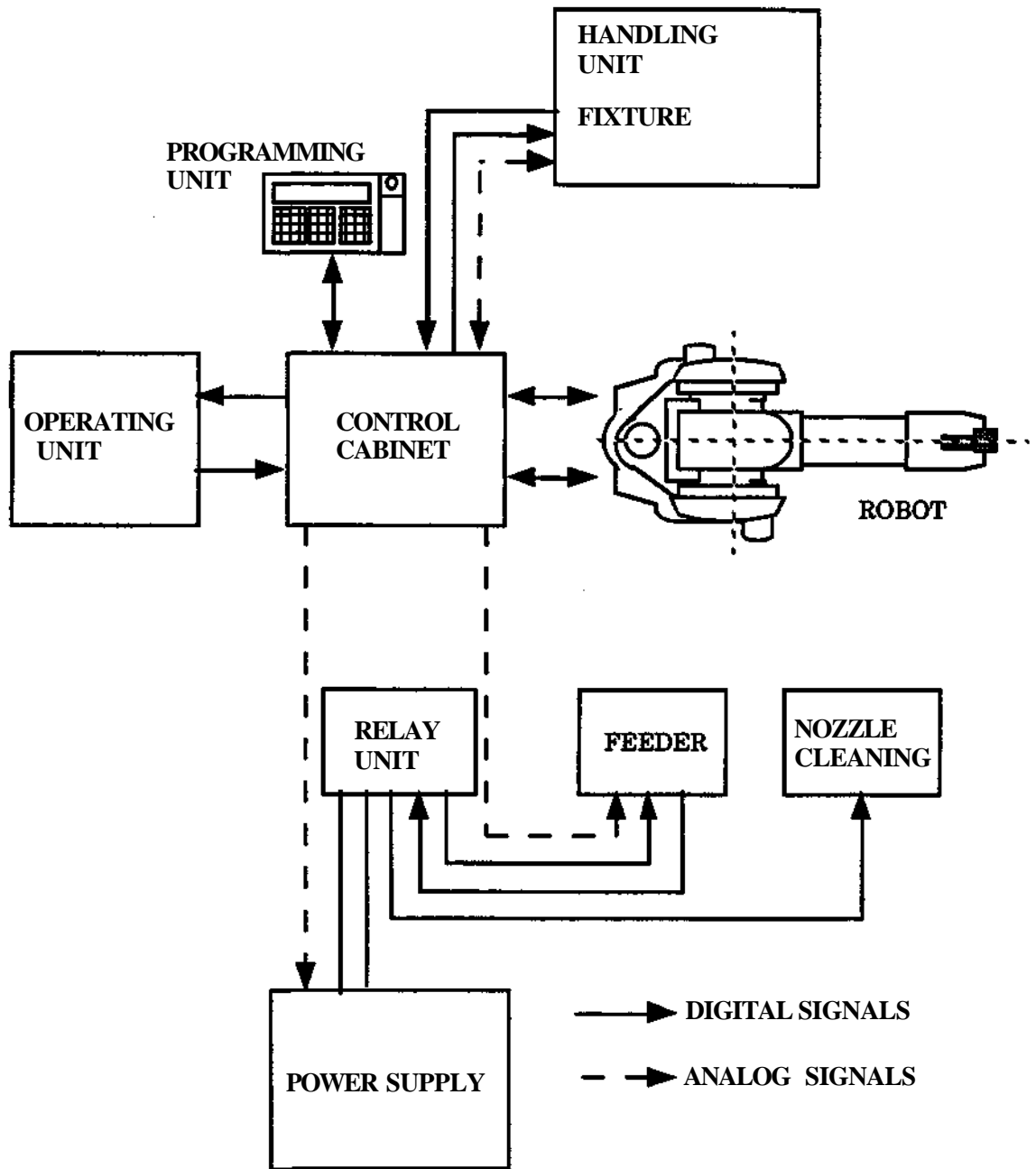
- Analog signals
 - reference voltage for control of the current source
 - reference voltage for control of the current wire feed unit
- Digital output signals
 - current source, on/off
 - wire feed unit, on/off
 - protective gas, on/off
 - spatter cleaning, on/off
 - contour following active (Applies only under certain conditions.)
 - right edge of joint (Applies only under certain conditions.)
 - left edge of joint (Applies only under certain conditions.)
- Digital input signals
 - current indication
 - supervision gas/coolant
 - ready signal (workpiece fixed)
 - command, block welding process
 - command, block weaving
 - command, manual wire feed

The signal exchange with the positioning equipment can vary considerably depending on the type of positioner (servo powered or indexing) and should be determined for each individual case.

All programming and testing of robot programs is performed from the programming unit. Normal production execution in the station is controlled from the station operator's panel which should be easily accessible to the operator.

The stations operator's panel can normally contain controls for:

- Program start
- Ready signal (workpiece fixed)
- **Program** stop
- Blocking of welding process and weaving
- Manual wire feed
- Emergency stop



REGISTER (AW)**Means:**

Control of program or peripheral equipment is performed on the basis of a value in a number register.

Facts:

The system contains 120 numerical registers, number 0 -119, where one value at a time can be stored. When a new value is stored, the previous value is written over. Permitted values are from -32 768 to +23 767.

During program execution, a numerical value can:

- Be stored directly.
- Be accessed from a digital or an analog input.
- Be sent to the peripheral equipment on digital or an analog output.
- Be compared with another value in a jump instruction, see "Jump within the program".
- Added or subtracted.
- Be a:
 - Program number at call of a program, see "Program change".
 - Number in a position register when moving to a stored position, see "Position register".
- Specify which module in a pattern program is to be called, see "Program structure".

It is also possible to check manually and if necessary, to change the value in one register at a time.

Used:

The register instruction itself is used for:

- Reading the value into a number register directly or with the help of the contents in other registers.
- Accessing a value from a port.
- Transmitting a value from a register to a port.

The manual function is used in checking and editing the program.

Executed:

The robot reacts immediately as soon as a manual or a programmed command, as described above, is concluded.

Interface to welding equipment

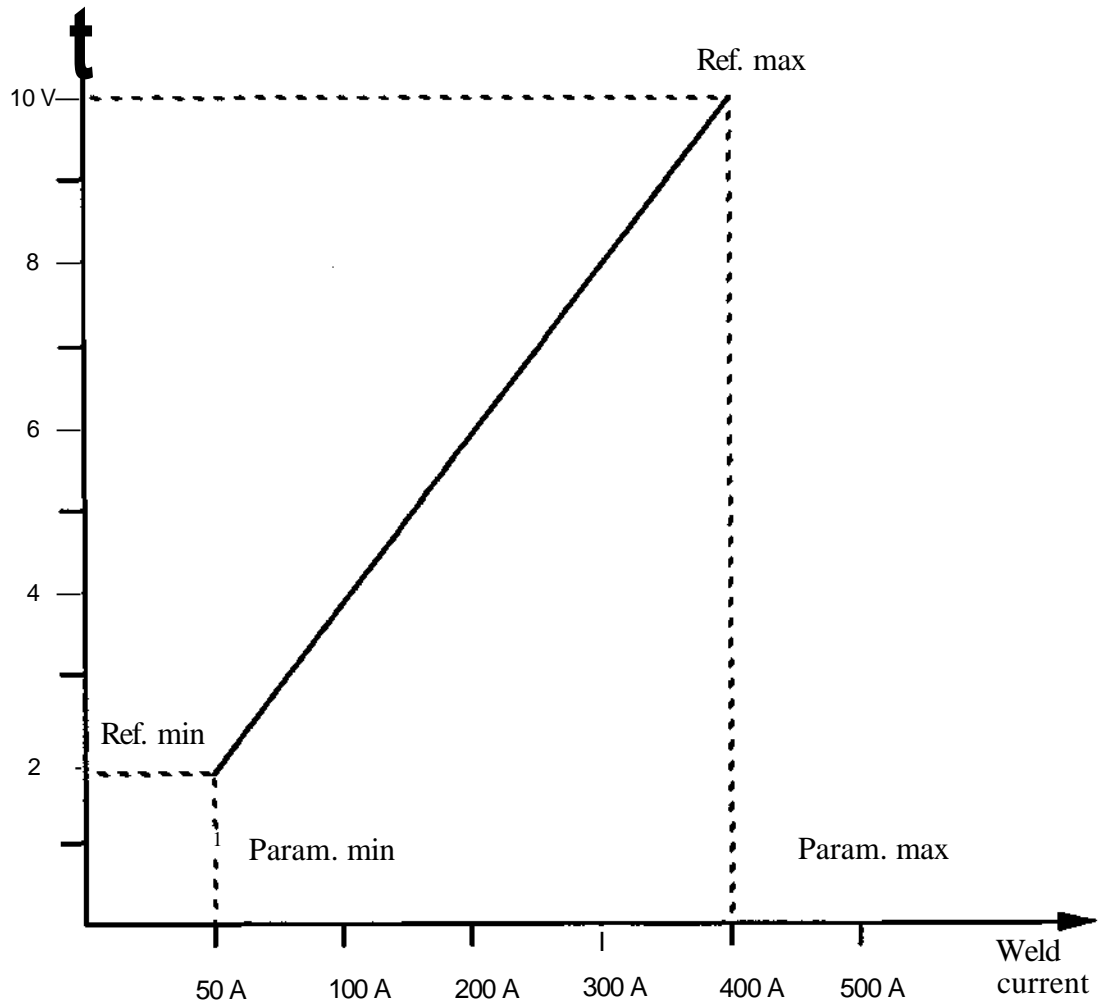
The robot is provided with an interface for communication with the welding equipment. A survey of the signals concerned is given below. More detailed instructions with respect to connection, function parameters, etc., are given in the Installation S3.

Analog outputs

The system is provided with two analog outputs for control of the power source (Port 21) and the wire feed unit (Port 22). The parameter output to the source -voltage- is expressed in volts. The parameter output to the wire feed unit is selected using a function parameter (see Installation S3) and can be either current (expressed in amperes) or wire speed (expressed in meter/minute or inch/minute). These outputs are normally dedicated to the functions specified above (via the integrated welding process control) but they can also be programmed freely (via I/O instructions).

The relation between the parameter value, programmed via the programming unit and the reference voltage obtained from the analog outputs is determined by function parameters. See also figure below.

Ref. voltage
from analog
output



Example of relation between programmed value (current in A) and reference voltage from analog output.

The function can be specified via function parameter definition of:

- Min/max values for reference
- Min/max values for parameter value

Note!

The functions described below are completely integrated in the robot system and no programming of these is necessary.

Digital inputs

The robot is provided with five special digital inputs for the arc welding application.

- Current indication (INPUT 7)

The input is used for supervision of the ignition sequence and the continuous welding and is connected to the current relay in the power source.

The signal is to be active for at least 50ms during the ignition procedure for the arc to be considered to be struck. If the arc has not struck within the weld start max time specified in the start data, an error indication is presented.

If during continuous welding the signal is absent for longer than 150 ms, the arc is considered to have extinguished and an error is presented.

In the event of malfunction the execution of the program is halted and an error printout is presented.

- Gas/coolant (INPUT 6)

The input is used for supervision of protective gas and/or liquid cooling and is connected to the flow monitors in the welding equipment.

The input is monitored at weld-start and continuously during the welding process.

If the signal is absent, execution of the program is halted and an error is presented.

- Manual wire feed (INPUT 8)

The input is used when the wire is fed forward manually, e.g. when a new wire is introduced.

The following takes place when the input is set:

- Output 8 (wire feed unit on) is set
- 30 % of max reference voltage is issued to the wire feed unit

The command is only accepted when the program is not execution.

- Blocking of welding process (INPUT 9)

The input is used to block the welding process manually during, for example, testing of the robot movements.

- If the input is set during program execution, any welding in progress will be halted as from the next WELD instruction.

- Blocking of weaving (INPUT 10)

The input is used to block a superimposed weaving motion manually during, for example, testing of the robot movements.

Digital outputs

The robot is provided with three special digital outputs for the arc welding application.

- Switch-on of power source (OUTPUT 7)

The output is set when the gas pre-flow time has expired. The signal then remains active during welding until the weld end procedure has started. During the conclusion procedure, the output is reset during the cooling time and set again during the crater-fill time and the burn-back time.

- Switch-on, wire feed unit (OUTPUT 8)

This output has the same function as "Switch-on, power source" except for during the burn-back time when it is cleared.

- Switch-on, protective gas (OUTPUT 9)

The output is set approximately 250 ms before the robot is in position for weld-start. It remains active until the gas post-flow time has expired.

- Contour following active (OUTPUT 19)

The output is set to logical "1" when the welding arc is energized. The output setting will remain until just before the welding is interrupted. The output is valid when the WELD-instruction is executed with automatic contour following.

- Right edge joint (OUTPUT 17)

The output will be set to logical "1" to indicate the right limit of the weaving during contour following.

- Left edge joint (OUTPUT 18)

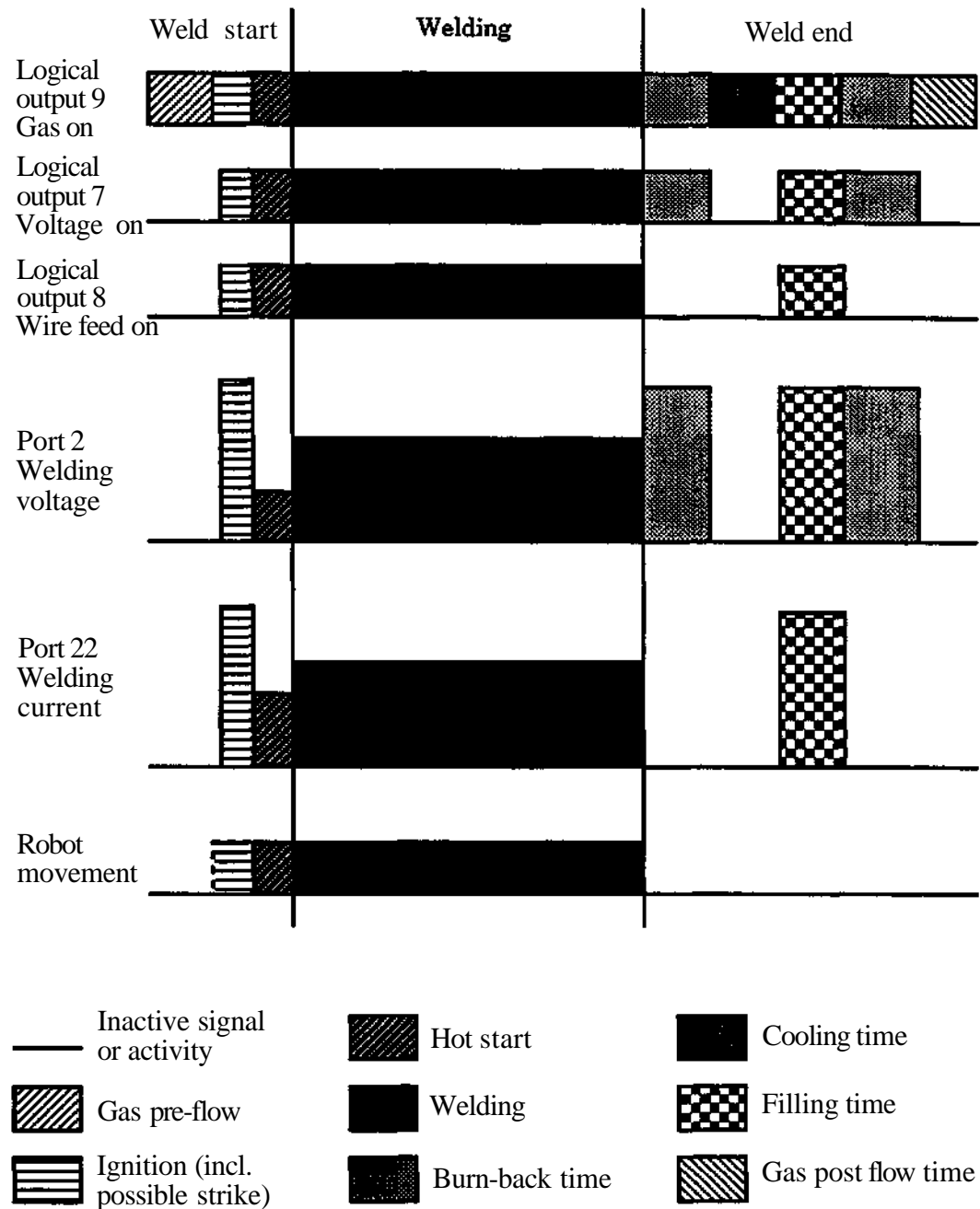
The output will be set to logical "1" to indicate the left limit of the weaving during contour following.

12 Arc welding

Arc welding-I/O is primarily used with ESAB welding equipment and positioner controlled with external axes and common drive unit.

A description of the principles of three arc welding functions is given below:

Welding cycle during three instructions



Servo controlled positioner with common drive unit

The external axis can be distributed to different stations with the help of the function STATION so that it becomes possible to activate these in suitable groups.

The stations can be activated/deactivated independently, except when axes share a common drive unit. An input and an output are connected to each station to permit activation of the station and to check the status of the station respectively.

Example:

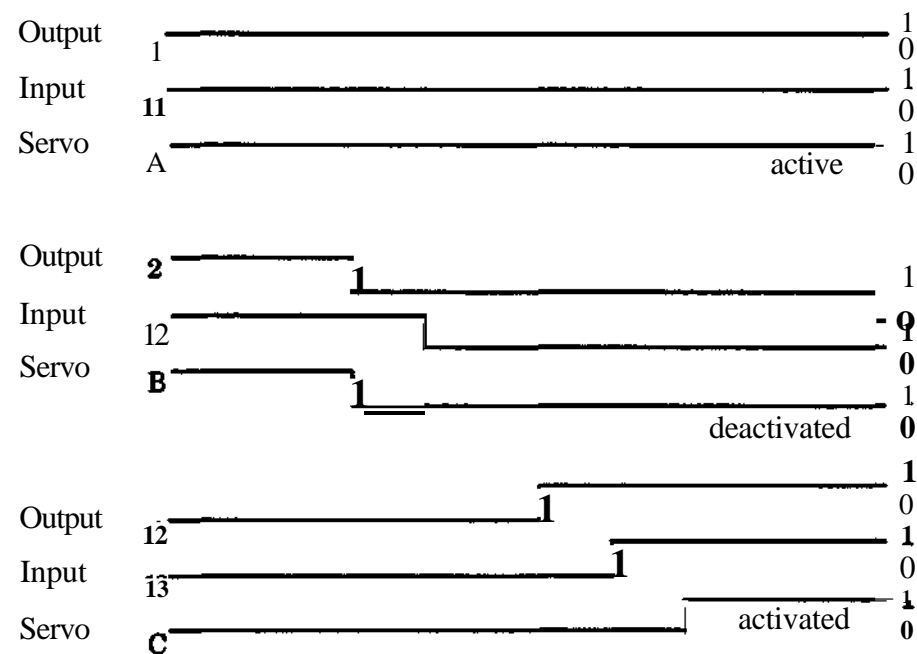
STATION A	INPUT 11	OUTPUT 1	and	AXES 7
STATION B*	INPUT 12	OUTPUT 2	and	AXES 8, 9
STATION C*	INPUT 13	OUTPUT 12	and	AXES 10, 11
STATION D	INPUT 14	OUTPUT 13	and	AXES 12

Axes 8 and 10 respectively 9 and 11 share two drive units, which means that the stations B and C cannot be active simultaneously. This is marked with a "*".

In this case the synchronization order would be the following:

1. C +robot
2. B + A + D +robot
3. B + D is being deactivated so that only A is activated after the synchronization.

A switchover from station B to station C is illustrated by the following time diagram:



Signals, servo controlled positioner with common drive unit

Logical input/output	Function
Outputs 1-128	An output used to activate/deactivate the drive unit of a station. The servo-control is disconnected before the output is reset to zero to deactivate the drive unit.
Inputs 11-16	The acknowledgement that a station is activated. The servo control is connected when the signal has been received.
Only one input and one output is used in each station in accordance with the function parameter STATION.	
Non-servo-controlled positioner A non-servo-controlled positioner can be run with the help of ports 70 and 80. A description is given in section 12.5.2.	

12.3**Program structure****Robot program construction**

A robot program for a robot welding station has in principle, the following construction elements

- **Welding data**, special data fields containing parameter values for control of the welding procedure. Welding data is programmed under the control button MANUAL, for preference, separately from the programming instructions themselves.
- **Main program**, includes basic data such as basic speed, TCP and coordinate type, welding and positioning instructions and other instructions of a general nature as required. The system welding instructions are of a combination type, i.e. they command positioning of the welding gun and call of the appropriate welding data selected. The welding instructions are programmed under the control button PROCESS (P) and may be conveniently entered into a subprogram, a welding program.
- A series of subprograms, as required, for control of spatter cleaning equipment, positions, weaving, etc. The system has special instructions for checking and control of external equipment and spatter cleaning. These instructions are programmed under the control button PROCESS (P).

A good structure is of great importance in a robot program. It permits easier tracing of faults, amendments and additions.

12 Arc welding

Even if subprograms may be numbered optionally (except interrupt programs) it is recommended that subprogram groups be organized so that a certain type is always within the same number series as shown in the following table.

Program number	Program type
----------------	--------------

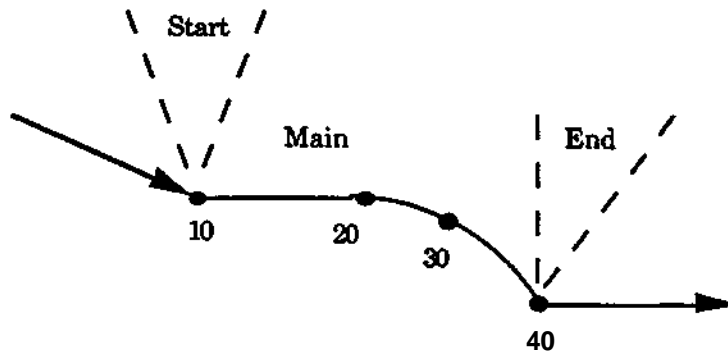
0	}	Main program
1		
‘	}	Interrupt programs
‘		
‘		
5	}	Welding programs
100		
‘		
‘		
199	}	External check program
200		
‘	}	Spatter cleaning program
‘		
‘		
299	}	Weaving program
300		
‘		
‘		
399	}	Other subprograms
400		
‘		
‘		
‘	}	
499		
‘	}	
500		
‘	}	
‘		

Building the robot program of these elements

- WELDING DATA
- MAIN PROGRAM
- SUBPROGRAMS

provides a good structure, and changes and additions are easily performed.

WELDING DATA is further divided into five different groups to further simplify programming and to increase the possibilities of flexible program construction.



- START DATA which contains parameter values for the weld start procedure.
- MAIN DATA which contains parameter values for the welding operation itself.
- END DATA which contains parameter values for the conclusion of the welding.
- WEAVE DATA, containing parameter values for automatic weaving.
- SENSOR DATA, containing parameter values for contour following.

The different types of DATA are called by the weaving instructions when executing the program.

- The instruction AWELD calls START DATA, MAIN DATA, WEAVE DATA and SENSOR DATA
- The instruction WEND calls END DATA.

Main principles for programming

Preparation of a robot program for arc welding should follow a predetermined plan, for example, as follows:

- Examine the workpiece concerned and determine the order in which the different welds are to be run. Determine also in which positions of the positioner the different joints are to be welded.
- Make an estimate of the parameter values for the different welding runs and program the necessary welding data.
- Program the necessary basic data in the main program.
- Program suitable sub-programs for, for example, spatter cleaning and control of the positioner.
- Program the robot motion with POS / AWELD / WEND instructions. The principle on which arc welding programming is based is:
 - Position the robot to the start or end point required.
 - Press the P-button.
 - Select the function required (AWELD or WEND)

At suitable places, CLEAN and EXTERNAL CONTROL (EXTC) instructions are programmed in accordance with the same principle .

- When the first version of the program is complete, a test run can be executed to permit correction of any errors. Test the robot motion first with the welding and any weaving function blocked. Trimming to obtain optimum welding data is performed with the help of the override function available by means of the programming unit joystick. (See further section 12.7.2)

12.4

Description and programming of welding data

Welding data; description

WELD DATA consists of special collections of data in which all parameter values necessary to control the welding process are assembled.

WELD DATA is of five different types:

- START DATA
- MAIN DATA
- END DATA
- WEAVE DATA
- SENSOR DATA

WELD DATA is programmed under the control button MANUAL by means of a dialogue procedure in a manner similar to that used for other programming. Parameter values are expressed in technical terms (V, A, mm/s, etc.) and it is easy to check and amend values programmed.

Restrike is defined under **START DATA**. It removes coating from the point of the weld wire by performing a zig-zag weaving at one point. The weaving movement is perpendicular to the welding direction in all cases except when the following point is a circle point. The weaving movement will then be perpendicular to the vector from the start point to the circle point. The weaving motion will continue until the arc is lit, then the movement will terminate and return to the middle line.

Restrike is defined by defining the amplitude and weaving time. If these are not given, the weld start will be performed without restrike.

WEAVE DATA is a data group which contains all data necessary for the weaving movement of the welding gun. They are:

Weave-type:

- Zigzag
- Triangular
- V-shape
- Wristweave

Amplitude: Total width of weaving movement. The weaving amplitude for the original *zig-zag* pattern is, as before, the peak to peak amplitude. With triangular and V-shaped weaving, the weaving amplitude is defined as the base of the isosceles triangle formed by the lines joining the weaving points, i.e. the distance between points 2 and 3 and between points 2 and 5 with triangular and V-shaped weaving respectively.

Cross time: The cross time is the time required for the movement, by the shortest path in the weaving pattern, between the points which limit the weaving. A specified cross time gives different weaving frequencies determined by the weaving pattern.

Dwell time Left: During the dwell time, the TCP is moved parallel with the direction vector from point 1 to point 2 with triangular and V-shaped weaving in accordance with the description below.

12 Arc welding

Dwell time Right:	During the dwell time, the TCP is moved parallel to the direction vector from point 3 to point 4 with triangular weaving and from point 5 to point 6 with V-shaped weaving in accordance with the description below.
Dwell time Middle:	During the dwell time, the TCP is moved parallel with the direction vector from point 5 to point 6 with triangular weaving. With V-shaped weaving, there is corresponding movement with both passages of the mean point, i.e. from point 3 to point 4 and from point 7 to point 8 in accordance with the description below.
Seam angle:	The seam angle is the opening angle of the triangular and V-shaped weaving. When the forward bias is not zero, the seam angle is defined as the opening angle of the weaving triangle projected on a plane at right angles to the direction vector.
Forward bias:	Forward bias is the component of the movement, parallel with the direction vector, which is subtracted from pos 1,2,3 and 4 with triangular weaving, and pos 1,2, 5 and 6 with V-shaped weaving in accordance with the description below.
Weaving angle:	The weaving angle is the angle between the TCP vector in the tool plane and the weaving vector, according to the weaving-coordinate system below.
Perpendicular weaving:	Determines whether the weaving direction is to be calculated at every step order (every 48:th milliseconds) or only for each AWELD instruction.

Note!

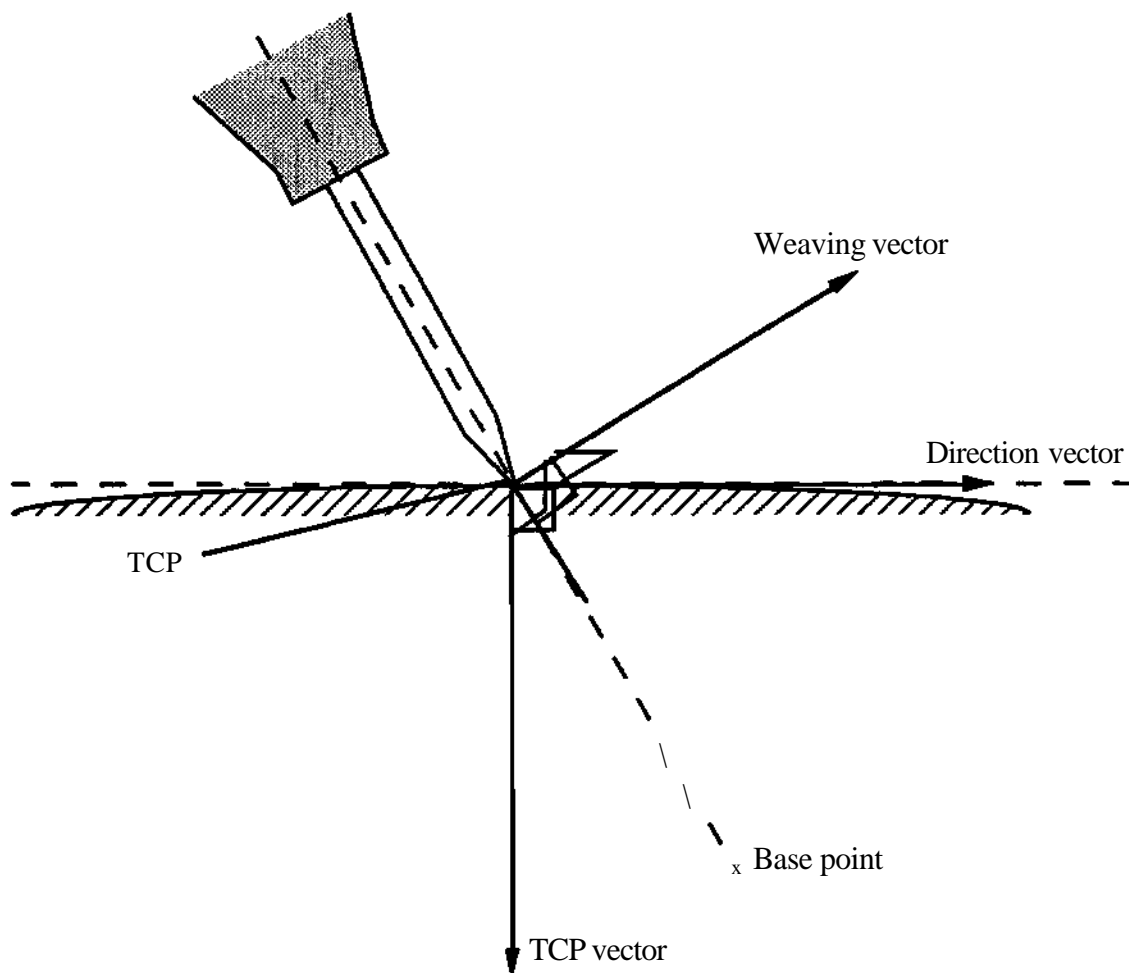
Dwell time left and right (and middle at V-shaped and triangular weaving) should not be chosen less than 0.05 s, besides when weaving with maximum weaving frequency. This dwell time is needed to obtain full amplitude at weaving.

Zig-zag, triangular and V-shaped weaving

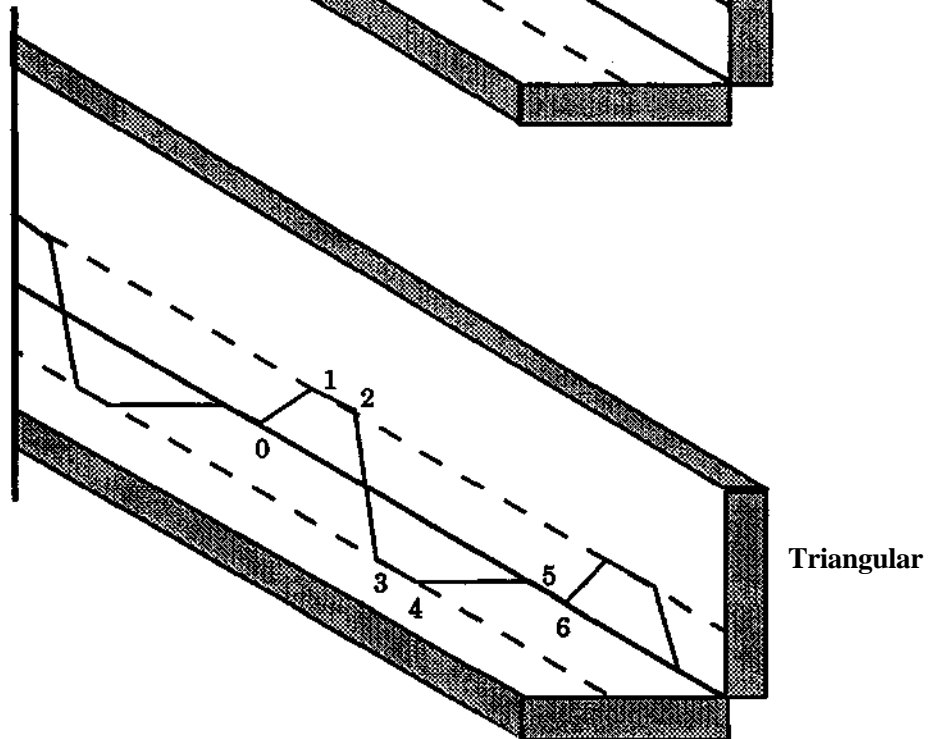
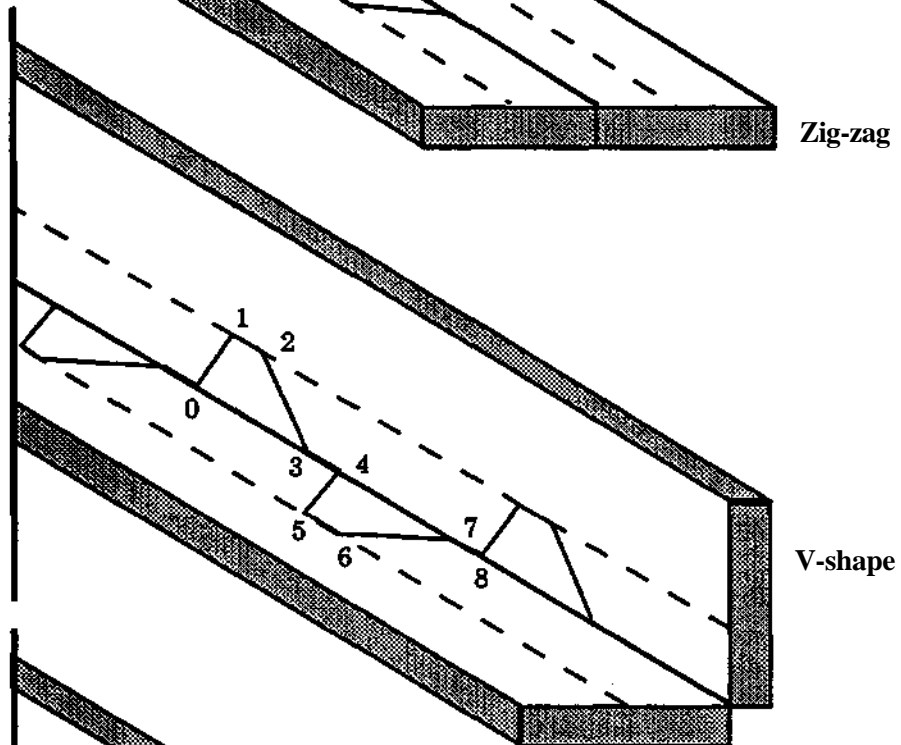
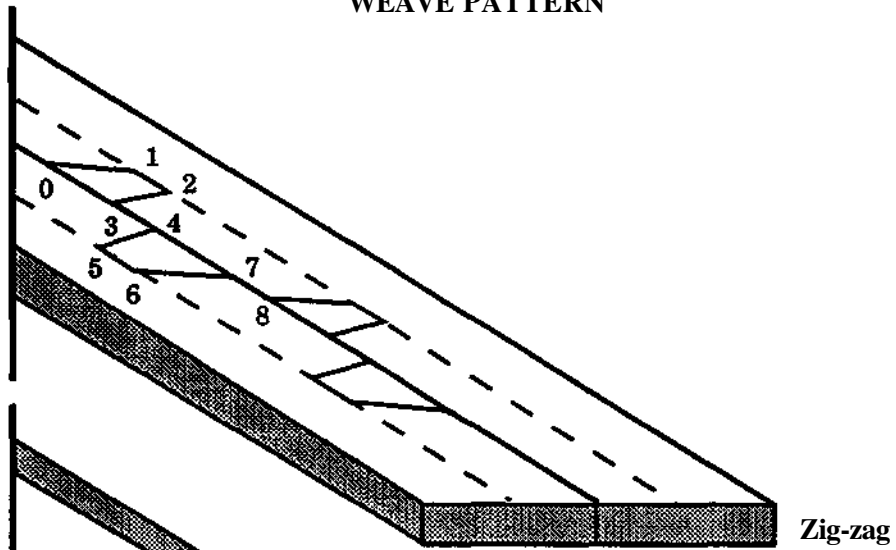
The weaving points which are superimposed on the process movement are described below in a coordinate system which is given by the direction vector, the weaving vector and the TCP vector.

12 Arc welding

The weave-coordinate system will adapt to the path so that the weaving will be carried out perpendicular to the path for velocities down to 2,7 mm/s. Below this velocity, the operator must himself check that the weaving is oriented correctly in relation to the path of the ordinary movement.



WEAVE PATTERN

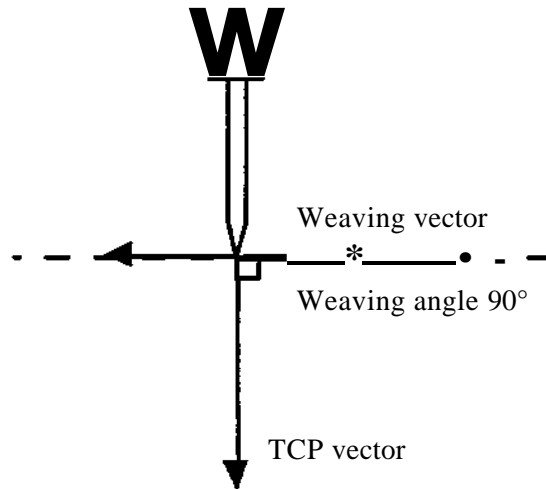


12 Arc welding

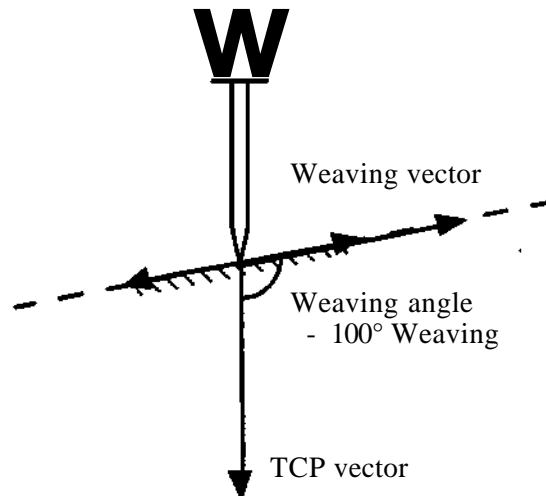
The tool direction defined by the TCP and base point, is always in the plane formed by the TCP vector and the direction vector in the weaving-coordinate system. The weaving vector is normally orientated perpendicular (90°) to this plane, which we call the tool-plane.

The direction of the weaving vector to the tool plane can be varied by help of the weaving angle parameter according to the following figures.

Zig-zag weaving, seen from the front.



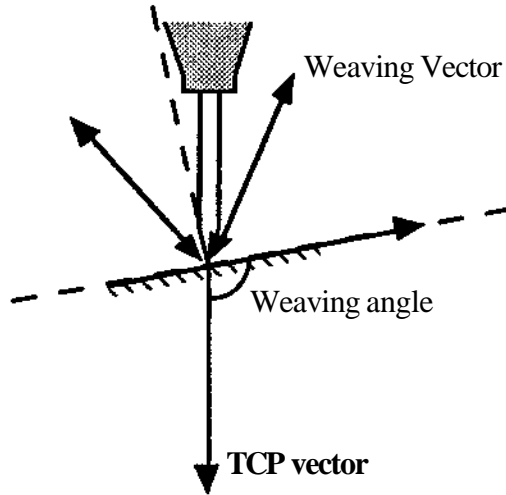
Zig-zag weaving, seen from the front.



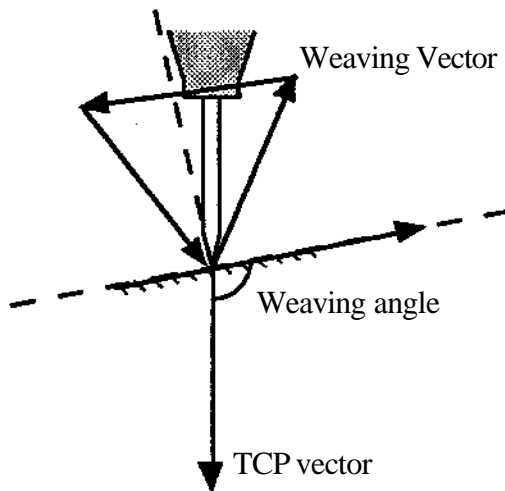
12 Arc welding

The three dimensional patterns, V-shaped and triangular weaving, are always performed symmetric to the normal of the weaving vector, as shown below.

V-shaped weaving, seen from the front.



Triangular weaving, seen from the front.



Weaving points

Definitions:

V : Dwell time Left
 Vv : Dwell time Right
 Vh : Dwell time Middle
 Am: Amplitude
 L : Forward bias
 H : Process velocity
 F : Seam angle
 Pn (m): Weaving point number n component m
 m=1: direction vector
 m=2: weaving vector
 m=3: TCP vector

Triangular weaving

$P(X1) = 0$	$P1(1) = -L$
$P(K2) = 0$	$P1(2) = A/2$
$POO = 0$	$P1(3) = A / (2 * \tan(F/2))$

$P2(1) = P1(1) + W * H$	$P3(1) = P2(1)$
$P2(2) = P1(2)$	$P3(2) = -A/2$
$P2(3) = P1(3)$	$P3(3) = P2(3)$

$P4(1) = P3(1) + Vh * H$	$P5(1) = P4(1) + L$
$P4(2) = P3(2)$	$P5(2) = 0$
$P4(3) = P3(3)$	$P5(3) = 0$

$P6(1) = P5(1) + Vm * H$
 $P6(2) = P5(2)$
 $P6(3) = P5(3)$

Output signals:

Weaving points PO PI P2 P3 P4 P5 P6

Output signals
Left

Output signals
Right

The output signals are valid at least 15 ms.

12 Arc welding

V-shaped weaving (also Zig-zag weaving, but then with a seam angle of 180°)

$$\begin{aligned} P0(1) &= 0 & P1(1) &= -L \\ P0(2) &= 0 & P1(2) &= A/2 \\ P0(3) &= 0 & P1(3) &= A/(2*\tan(F/2)) \end{aligned}$$

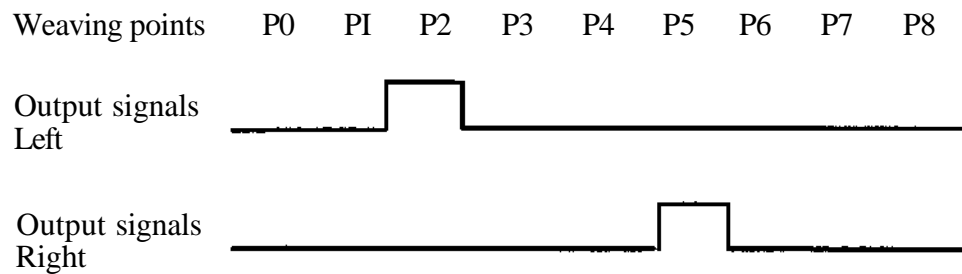
$$\begin{aligned} P2(1) &= P1(1) + W * H & P3(1) &= P2(1) + L \\ P2(2) &= P1(2) & P3(2) &= 0 \\ P2(3) &= P1(3) & P3(3) &= 0 \end{aligned}$$

$$\begin{aligned} P4(1) &= P3(1) + V_m * H & P5(1) &= P4(1) - L \\ P4(2) &= P3(2) & P5(2) &= -A/2 \\ P4(3) &= P3(3) & P5(3) &= A/(2*\tan(F/2)) \end{aligned}$$

$$\begin{aligned} P6(1) &= P5(1) + V_h * H & P7(1) &= P6(1) + L \\ P6(2) &= P5(2) & P7(2) &= 0 \\ P6(3) &= P5(3) & P7(3) &= 0 \end{aligned}$$

$$\begin{aligned} P8(1) &= P7(1) + V_m * H \\ P8(2) &= P7(2) \\ P8(3) &= P7(3) \end{aligned}$$

Output signals:



The output signals are valid at least 15 ms.

Wrist weaving

Wrist weaving is a special type of weaving for axis 6. It enables weaving in confined spaces where weaving with axes 1-3 is impossible. Wrist weaving can also be used when a higher weaving frequency is required as the dynamic mass is smaller.

Designations

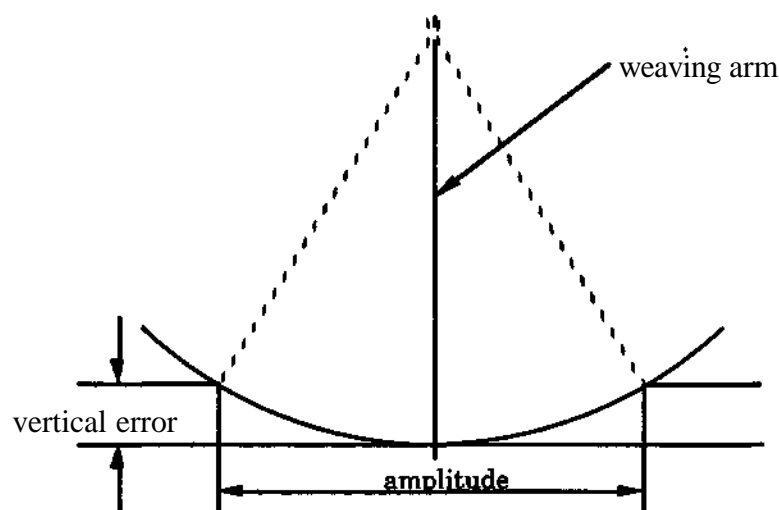
Weaving arm is an arm which extends from the TCP to the weaving axis. The arm is perpendicular to the weaving axis.

Weaving plane

Weaving plane is the plane in which the weaving arm moves.

The wrist weaving is superimposed on the ordinary movement, but there is no vertical error compensation. See the figure below. To reduce the vertical error, the weaving arm should be relatively large and the amplitude small.

The robot operator must check that the weaving plane is oriented correctly in relation to the ordinary movement. The rules which apply otherwise to the other types of weaving apply also to the wrist weaving.



A corner path (parabolic path) is generated at a corner for normal movements, but not for WEAVE data movement. Weaving starts when entering the zone of the start position and ends when entering the zone of the finish position.

If the weld data is changed at an intermediate position a fine point is automatically defined. To avoid a temporary halt in robot movement at the fine point the programmer should change the position definition to a PATH or CORNER1.

Weaving at maximum weaving frequency

The parameters should be chosen as below when weaving at maximum weaving frequency is used.

Zig-zag, V-shaped and triangular weaving (maximum weaving frequency 5 Hz)

Cross time = 0.1 s
 Dwell time right = 0 s
 Dwell time left = 0 s
 Dwell time middle = 0 s

Wrist weaving (maximum weaving frequency 10 Hz)

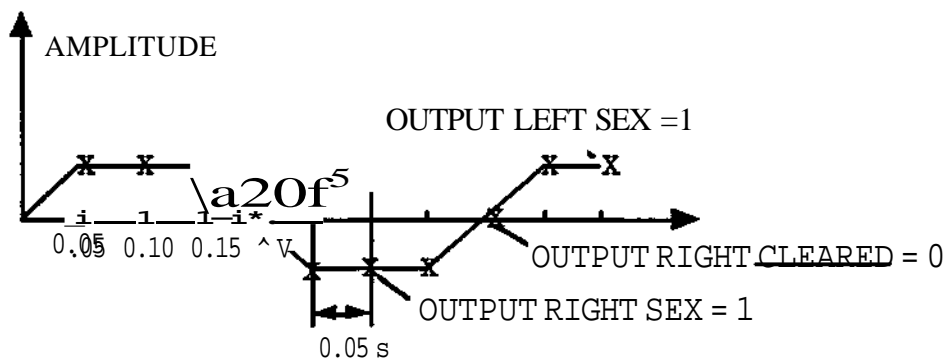
Cross time = 0.05 s

Dwell time right = 0 s

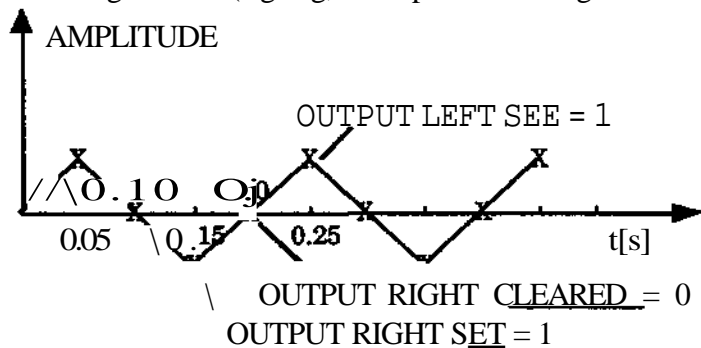
Dwell time left = 0 s

Note! Dwell time left and right (and middle at V-shaped and triangular weaving) should not be chosen less than 0.05 s, besides when weaving with maximum weaving frequency. This dwell time is needed to obtain full amplitude at weaving.

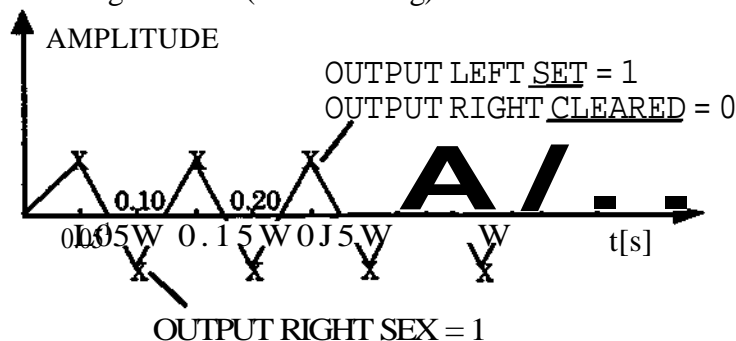
The handling of weaving outputs for contour following (output 17 and 18) are also affected when the dwell times are set to 0 s for weaving with maximum weaving frequency. These outputs are therefore set 0.05 s earlier than normal to be able to be set and cleared in pace with the maximum weaving frequency (see figures below).



Weaving at 5 Hz (zig-zag, V-shaped and triangular weaving)



Weaving at 10 Hz (wrist weaving)



Sensor data

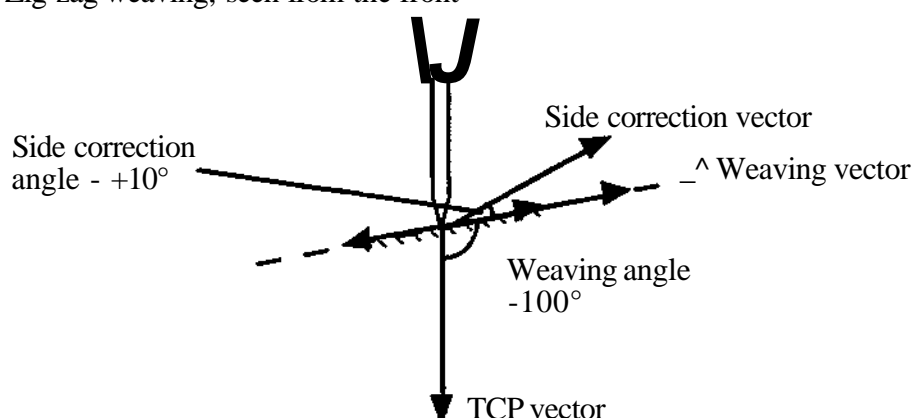
Sensor data is a collection of data containing necessary data for definition of contour following during AWELD. They are:

Side corr. sensor no.:	The number of the sensor used for side correction.
Side corr. sensor bias:	BIAS in percent of the sensor's total working range. 50 % indicates sensor in the centre of its working range.
Heightcorr. sensor no.:	The number of the sensor used for height correction.
Heightcorr. sensor bias:	BIAS in percent of the sensor's total working range. 50% indicates sensor in the centre of its working range.
Side corr. angle:	The angle between the weaving vector and the side correction vector at simultaneous contour following and weaving.

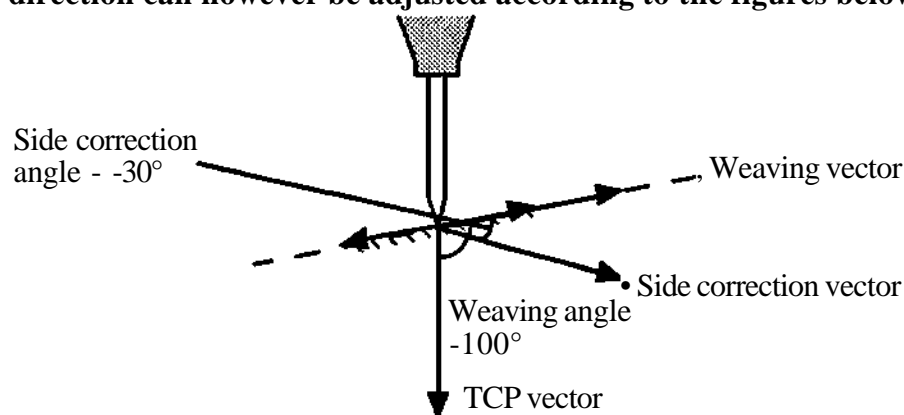
Normally the weaving vector corresponds to the side correction vector, which means that the side correction angle normally is 0° . The direction can however be adjusted according to the figures below.

The height correction vector corresponds to the TCP vector in both figures.

Zig-zag weaving, seen from the front



Zig-zag weaving, seen from the front correction angle normally is 0° . The direction can however be adjusted according to the figures below.



12 Arc welding

Fifty different collections of each MAIN DATA type can be defined.

Twenty different collections of each WEAWE DATA type can be defined.

Ten different data collections of each of the other WELD DATA types can be defined.

The table below shows the parameter values which can be programmed.

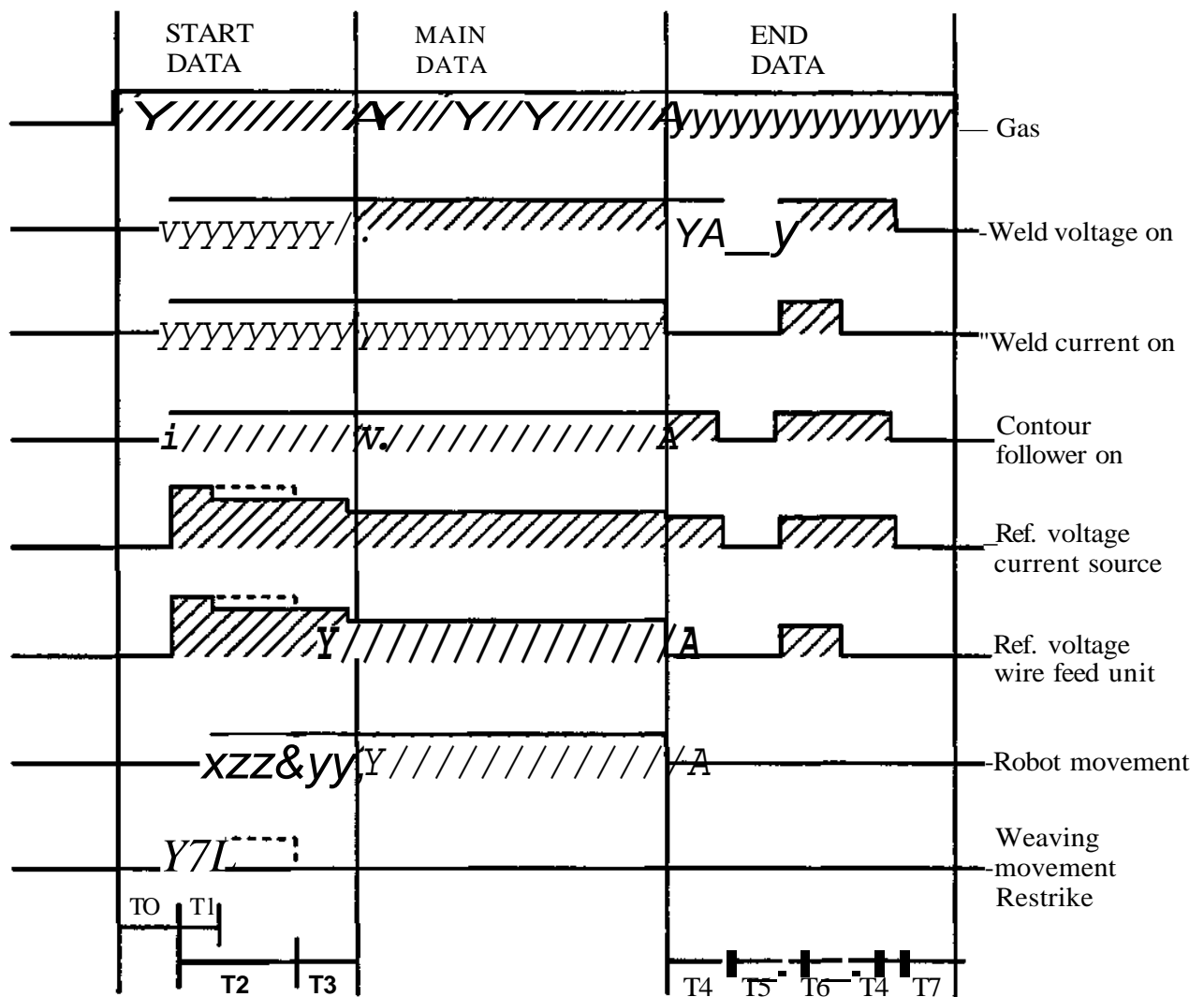
Current is replaced by wire speed if wire speed is selected in the system parameters, (see Installation S3).

Type	Parameter	Given in	Basic value	Min. value	Max. value	Resolution
Start data	Ignition voltage	Volt	0.0	-40.0	40.0	0.1
	Ignition current	Ampere	0.0	-400.0	400.0	0.1
	Ign. wire speed	m/min	0.0	-20.0	20.0	0.1
		inch/min	0	-787	787	1
	Gas pre-flow	seconds	0.00	0.00	99.99	0.01
	Hot start voltage	Volt	0.0	-40.0	40.0	0.1
	Hot start current	Ampere	0.0	-400.0	400.0	0.1
	Hot st. wire speed	m/min	0.0	-20.0	20.0	0.1
		inch/min	0	-787	787	1
	Hot start time	seconds	0.00	0.00	99.99	0.01
	Restrike amplitude	millimeter	0.0	0.0	99.99	0.1
	Restrike weave time	seconds	0.00	0.00	99.9	0.01
	Weld st. max time	seconds	2.00	0.50	99.99	0.01
Main data	Weld voltage	Volt	0.0	-30.0	100.0	0.1
	Weld current	Ampere	0.0	0	1000.0	0.1
	Weld wire speed	m/min	0.0	0.0	50.0	0.1
		inch/min	0	0	1968	1
	Weld speed	mm/s	0.0	0	200.0	0.1
		inch/min	0	0	476	1
End	End voltage	Volt	0	-40.0	40.0	0.1
	End current	Ampere	0	-400.0	400.0	0.1
	End wire speed	m/min	0	-20.0	20.0	0.1
		inch/min	0	-787	787	1
	Gas post-flow time	seconds	1	0	99.99	0.01
	Burn back time	seconds	0.05	0	99.99	0.01
	Cooling time	seconds	0	0	99.99	0.01
	Fill time	seconds	0	0	99.99	0.01
Weaving	Type	-	zig-zag	-	-	-
	Amplitude	mm	1.0	0.0	*) 99.99	0.01
	Cross time	seconds	1.00	0.00	99.99	0.01
	Dwell times:					
	left, right, center	seconds	0.05	0	99.99	
	Seam angle	degrees	90	10	170	1
	Forward bias	mm	0.0	0.0	99.9	0.1
	Weave angle	degrees	90	0	180	1
Sensor data	Perpendicular	Yes/No	Yes	-	-	-
	Side, corr. sensor no.	-	0	0	16	1
	Height, corr. sensor no.	-	0	0	16	1
	Side, corr. sensor bias	%	50	0	100	1
	Height, corr. sensor bias	%	50	0	100	1
	Side corr. angle	degrees	0	-90	90	1

Note! Ignition, hot start and end values for voltage and current/wire speed are given as differences relative to corresponding Main data.

*) IRB 1500: ZIG-ZAG, Triangular, V-shape maximum 16 mm, Wrist maximum 7 mm.

The following diagram shows the relation between different signals, times, etc.



TO = Gas pre-flow time

T1 = Ignition time (< 50 ms)

T2 = Maximum restrike time

T3 = Hot start time

T4 = Burn-back time

T5 = Cool time

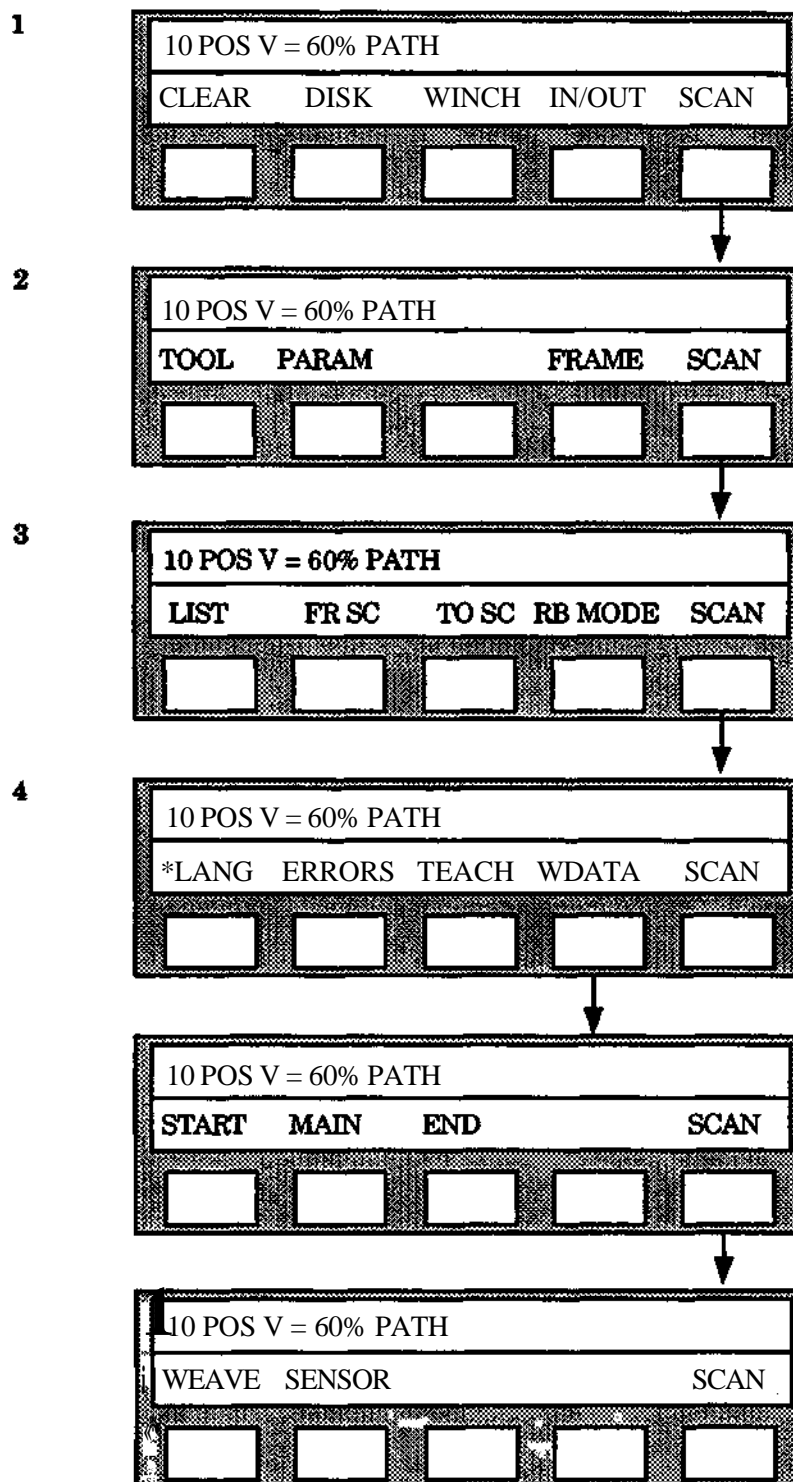
T6 = Filling time

T7 = Gas post-flow time

Programming of weld data

Welding data is programmed under the control button MANUAL. Begins with:

1. Press SCAN.
2. Press SCAN.
3. Press SCAN.
4. Press WDATA



After the above sequence, START DATA, MAIN DATA, END DATA, WEAWE DATA or SENSOR DATA can be programmed.

Programming START DATA

1. Press function button START.
2. Select the required START DATA number (1-10) with the numerical button set. Data numbers not used are marked on the display with -.
3. Press ENTER.
4. Write the ignition voltage difference value with the numerical button
5. Press ENTER. (* The value programmed is now shown on the upper line of the display. New values can be programmed directly if required.)
6. Press ENTER.
7. Write the ignition current/wire speed difference value required with the numerical button set.
8. Press ENTER. (* See point 5 above)
9. Press ENTER.
10. Write the gas pre-flow time value required with the numerical button set.
11. Press ENTER. (* See point 5)
12. Press ENTER.
13. Write the hot start voltage difference value required with the numerical button set.
14. Press ENTER. (* See point 5)
15. Press ENTER.
16. Write the hot start current/wire speed difference value required with the numerical button set.
17. Press ENTER. (* See point 5)
18. Press ENTER.
19. Write the hot start time required with the numerical button set.
20. Press ENTER. (* See point 5)
21. Press ENTER.
22. Write the restrike weaving amplitude required with the numerical button set.
23. Press ENTER. (* See point 5)
24. Press ENTER

25. Write the restrike weaving cross time required with the numerical button set.

26. Press ENTER. (* See point 5)

27. Press ENTER.

28. Write the weld start max time required with the numerical button set.

29. Press ENTER. (* See point 5)

30. Press ENTER.

N3!

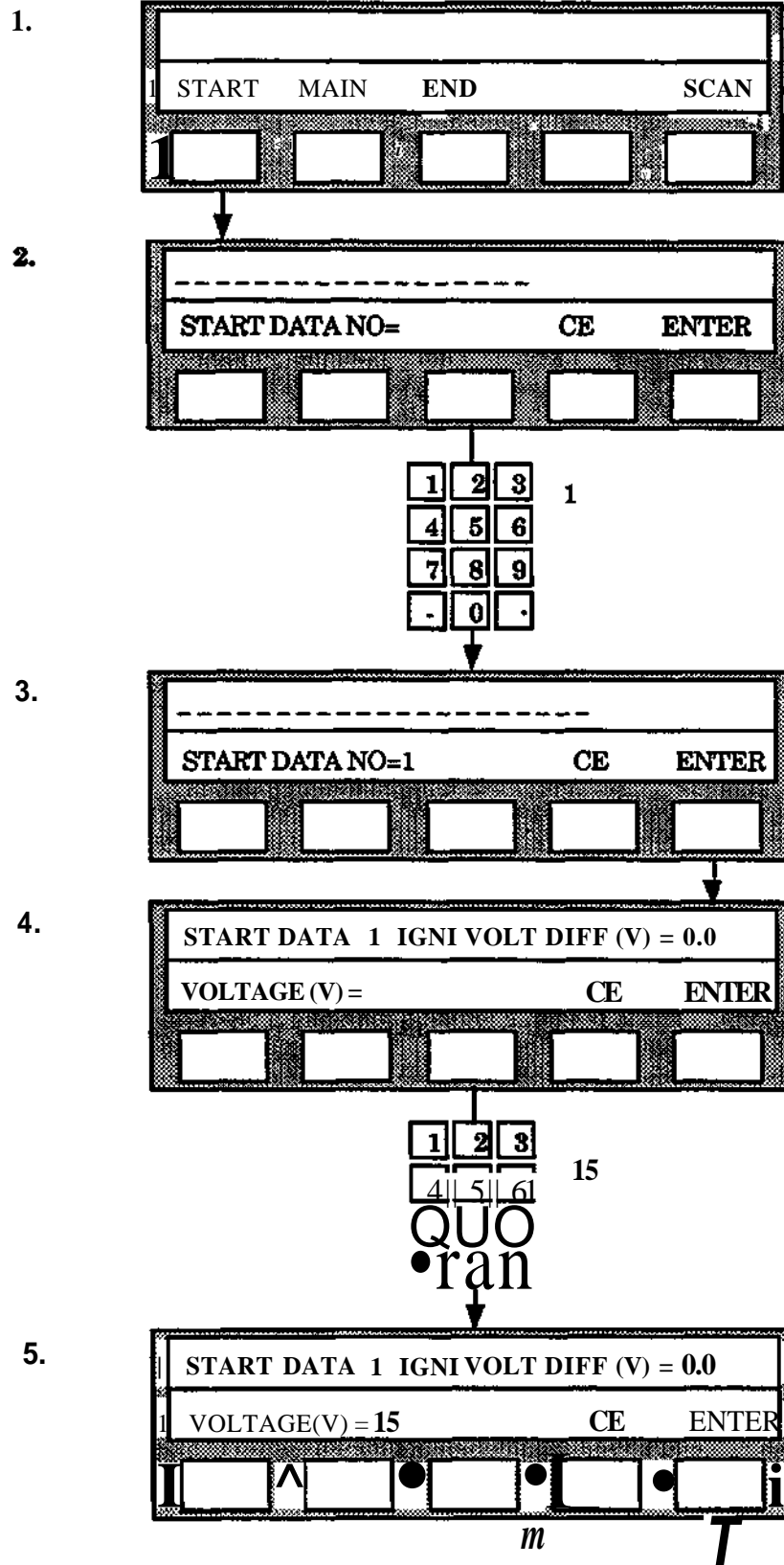
If it is decided to retain the basic value for a parameter, the sequence can be continued by pressing ENTER directly, instead of writing another numerical value.

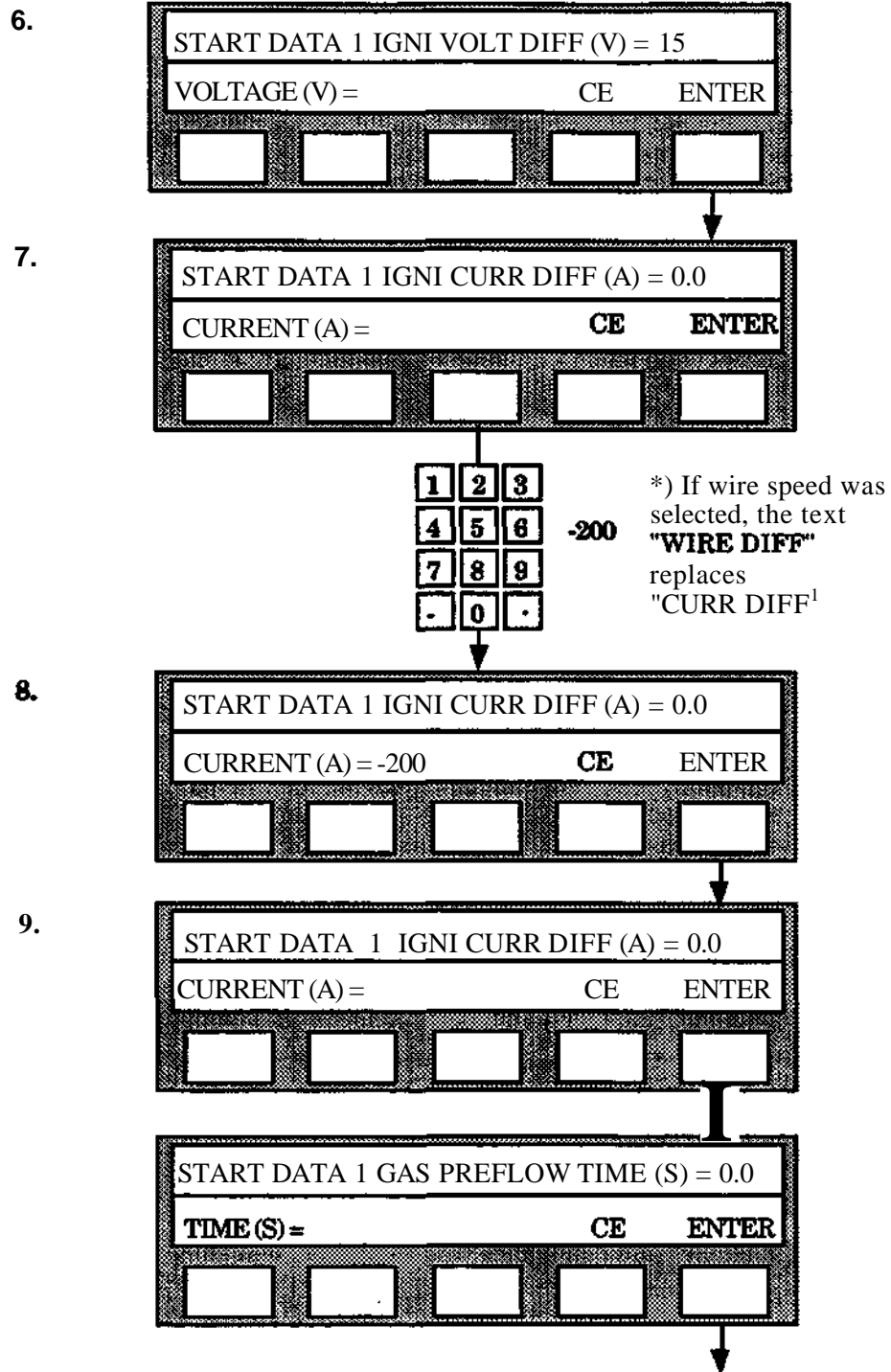
⌋

⌋

⌋

⌋





10.

1	2	3
4	5	6
7	8	9
.	0	.

1.5

11.

START DATA 1 GAS PREFLOW TIME (S) = 0.00				
TIME (S) = 1.5 CE ENTER				

12.

START DATA 1 GAS PREFLOW TIME (S) = 1.5 0				
TIME (S) = CE ENTER				

START DATA 1 HOT VOLT DIFF (V) = 0.0				
VOLTAGE (V) = CE ENTER				

13.

1	2	3
4	5	6
7	8	9
.	0	.

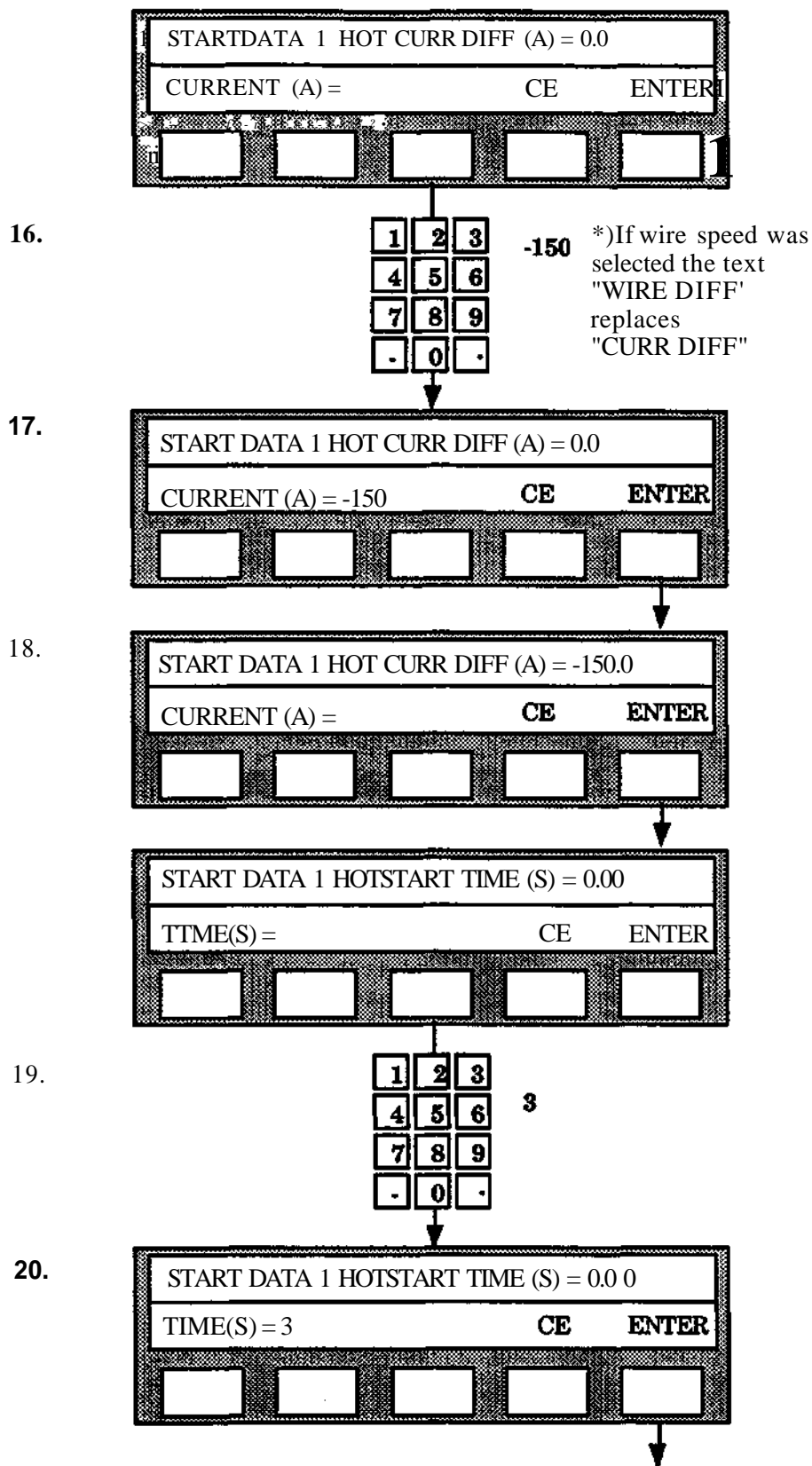
-10

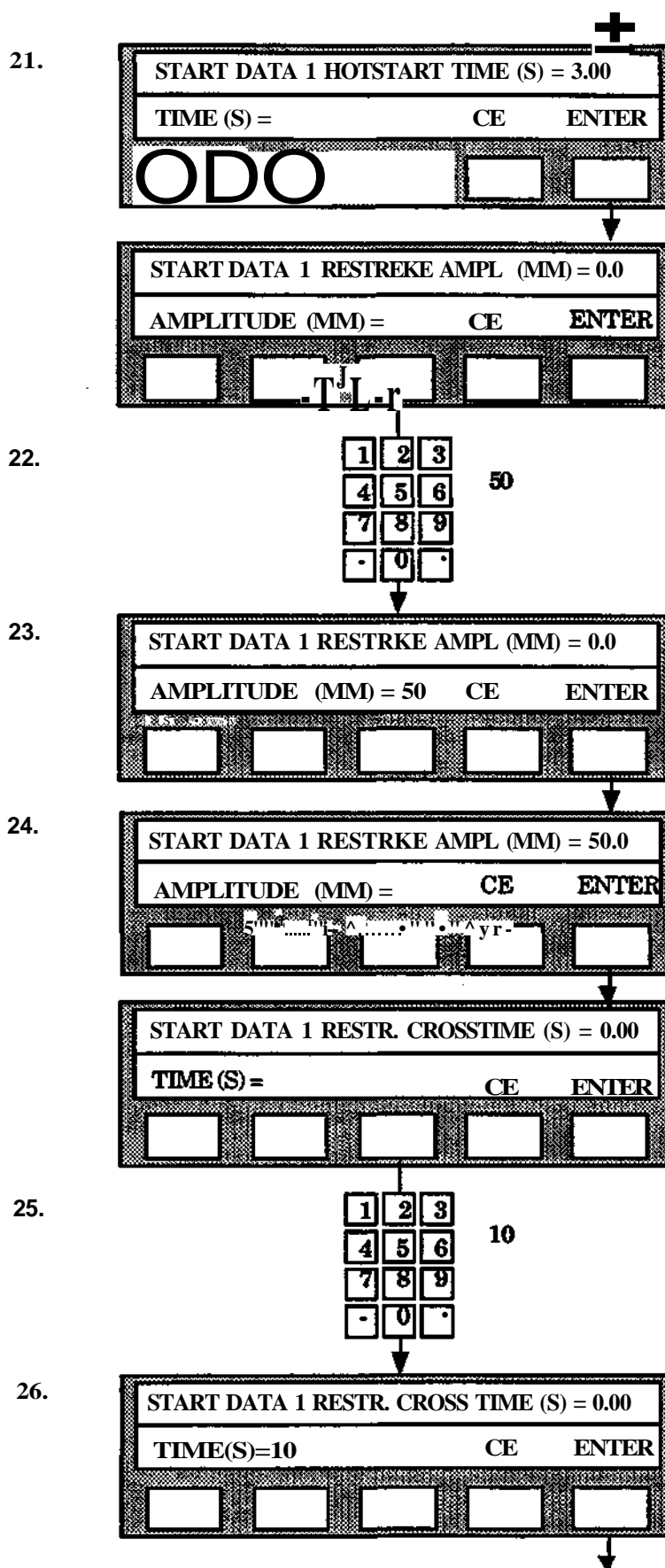
14.

START DATA 1 HOT VOLT DIFF (V) = 0.0				
VOLTAGE (V) = -10 CE ENTER				

15.

START DATA 1 HOT VOLT DIFF (V) = -10.0				
VOLTAGE (V) = CE ENTER				





27.

START DATA 1 RESTR. CROSSTIME (S) = 10.00				
TIME (S) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



28.

START DATA 1 AW START MAX. TIME = 2.00				
TIME (S) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3	1.5
4	5	6	
7	8	9	
.	0	.	



29.

START DATA AW START MAX. TIME = 2.00				
TIME (S) = 1.5		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



30.

START DATA AW START MAX. TIME = 1.50				
TIME(S) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



START	MAIN	END	SCAN	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Programming MAIN DATA

1. Press function button MAIN.
2. Select the MAIN DATA number required (1-50) with the numerical button set. Data numbers not used are indicated on the display with -.
3. Press ENTER.
4. Write the weld voltage value required with the numerical button set.
5. Press ENTER (* The value programmed is now shown on the display. New values can be programmed directly if so required.)
6. Press ENTER.
7. Write the weld current/wire speed value required with the numerical button set.
8. Press ENTER. (* See point 5)
9. Press ENTER.
10. Write the weld speed required with the numerical button set.
11. Press ENTER. (* See point 5)
12. Press ENTER.

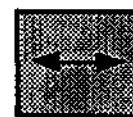
NJ3!

If it is decided to retain the basic value for a parameter, it is possible to proceed with the sequence by pressing ENTER directly instead of writing another numerical value.

1.

START	MAIN	END		

----->				
MAIN DATA NO =			CE	ENTER



use the SHIFT
button for further
data fields.

2.

1	2	3	1
4	5	6	
7	8	9	
.	0	.	

3.

----->				
MAIN DATA NO =			CE	ENTER

4.

MAIN DATA 1 VOLTAGE (V) = 0.0				
VOLTAGE(V) =			CE	ENTER

1	2	3	27
4	5	6	
7	8	9	
.	0	.	

5.

MAIN DATA 1 VOLTAGE (V) = 0.0				
VOLTAGE(V) = 27			CE	ENTER

6.

MAIN DATA 1 VOLTAGE (V) = 27.0

VOLTAGE (V) = CE ENTER

7.

1	2	3
4	5	6
7	8	9
-	0	.

150

*) If wire speed was selected the text "WIRE SPEED" replaces "CURRENT-"

8.

MAIN DATA 1 CURRENT (A) = 0.0

CURRENT(A)= 150 CE ENTER

9.

MAIN DATA 1 CURRENT (A) = 150.0

CURRENT (A) = CE ENTER

MAIN DATA 1 WELD VELOCITY (MM/S) = 0.0

VELOCITY (MM/S) = CE ENTER

10.

10

1	2	3
4	5	6
7	8	9
-	0	.

11.

MAIN DATA 1 WELD VELOCITY (MM/S) = 0.0				
VELOCITY (MM/S)= 10		CE	ENTER	

12.

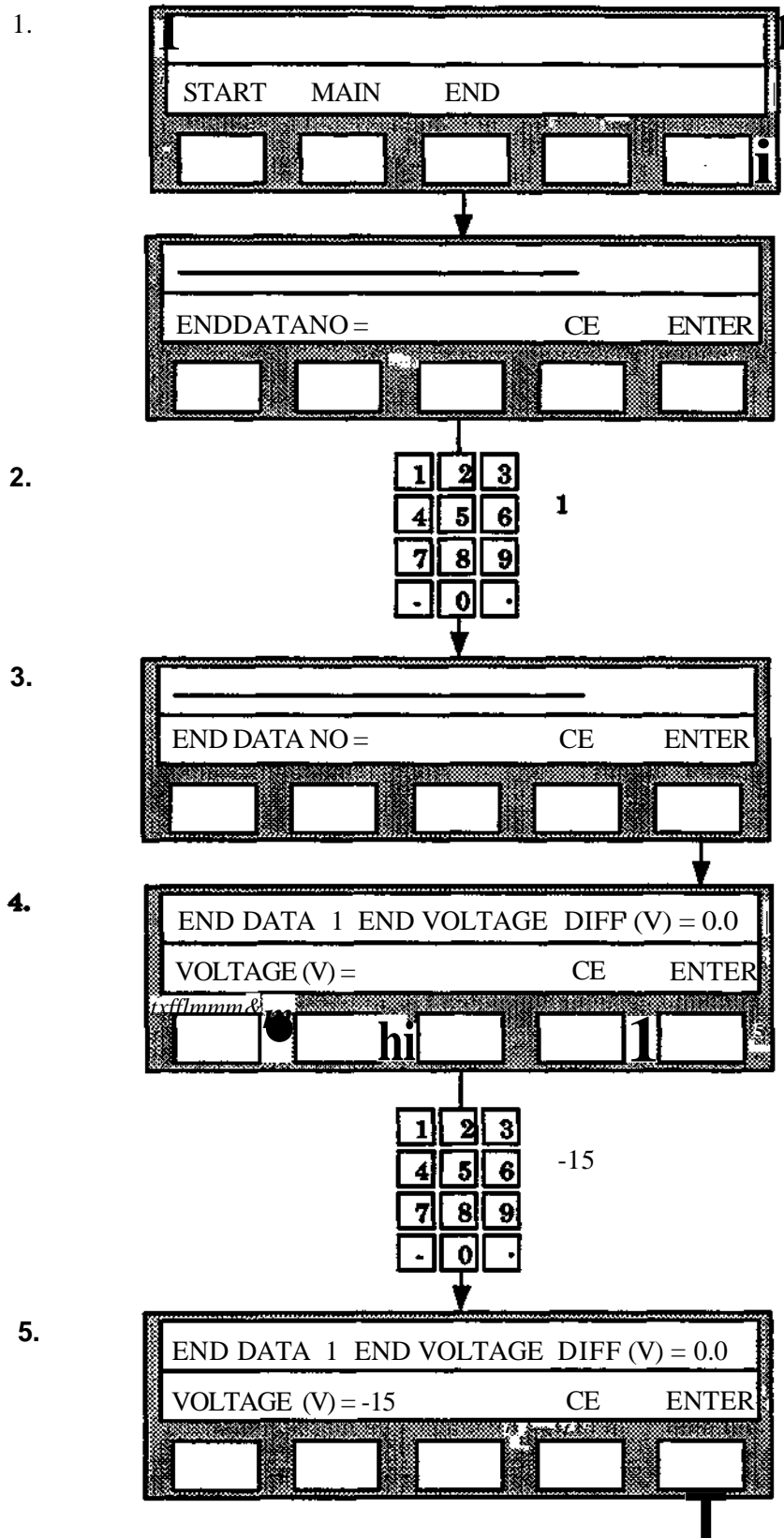
MAIN DATA 1 WELD VELOCITY (MM/S) =10.0				
VELOCITY (MM/S) =		CE	ENTER	

START	MAIN	END		

Programming END DATA

1. Press the function button END.
2. Select the END DATA number required (1-10) with the numerical button set. Data numbers not used are indicated on the display with -.
3. Press ENTER.
4. Write the end voltage difference value required with the numerical button set.
5. Press ENTER. (* The value programmed is now shown on the display. A new value can, if required, be programmed directly.)
6. Press ENTER.
7. Write the end current/wire speed difference value required with the numerical button set.
8. Press ENTER (* See point 5)
9. Press ENTER.
10. Write the gas post-flow time required with the numerical button set.
11. Press ENTER. (* See point 5)
12. Press ENTER.
13. Write the burn-back time required with the numerical button set.
14. Press ENTER (* See point 5)
15. Press ENTER
16. Write the cooling time required with the numerical button set.
17. Press ENTER (* See point 5)
18. Press ENTER
19. Write the fill time with the numerical button set.
20. Press ENTER (* See point 5)
21. Press ENTER

NJB!. If it is decided to retain the basic value of a parameter, the sequence can be continued by pressing ENTER directly instead of entering another numerical value.



6.

END DATA 1 END VOLTAGE DIFF (V) = -15.0				
VOLTAGE(V) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

END DATA 1 END CURRENT DIFF (V) = 0.0				
CURRENT (A) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

7.

1	2	3
4	5	6
7	8	9
-	0	.

-180

*) If wire speed was selected the text "WIRE DIFF" replaces "CURRENT DIFF"¹

8.

END DATA 1 END CURRENT DIFF (A) = 0.0				
CURRENT (A) = -180		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9.

END DATA 1 END CURRENT DIFF (A) = -180.0				
CURRENT (A) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

END DATA 1 GASPOSTFLOW TIME (S) = 1.00				
TIME(S) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

10.

1	2	3
4	5	6
7	8	9
-	0	.

1.5

11.

END DATA 1 GASPOSTFLOWTIME (S) = 1.00				
TIME (S) = 1.5				
			CE	ENTER

12.

END DATA 1 GAS POSTFLOW TIME (S) = 1.50				
TIME (S) =				
			CE	ENTER

END DATA 1 BURN BACK TIME (S) = 0.05				
TIME(S) =				
			CE	ENTER

13.

1	2	3
4	5	6
7	8	9
-	0	.

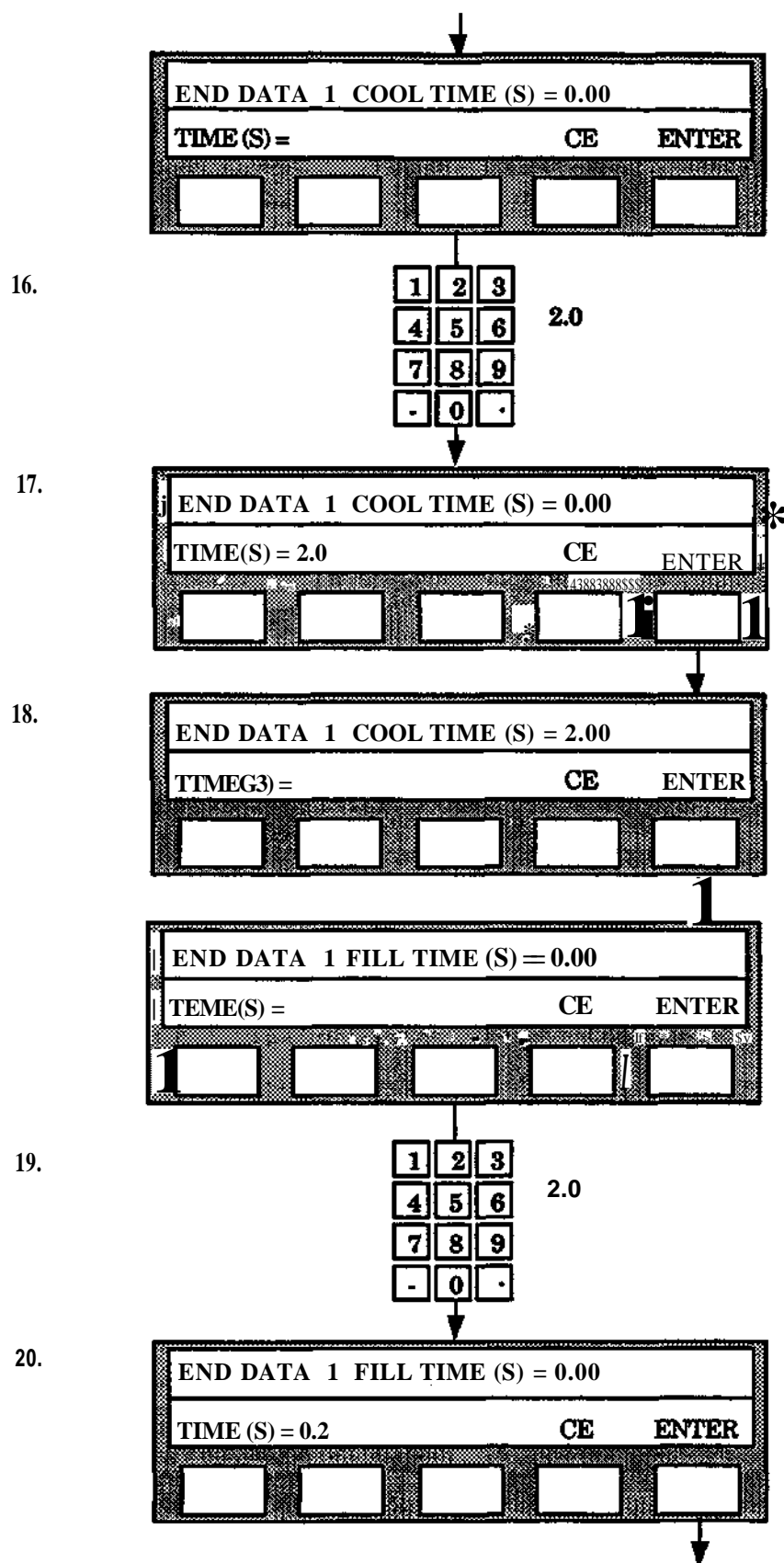
0.2

14.

END DATA 1 BURN BACK TIME (S) = 0.05				
TIME (S) = 0.2				
			CE	ENTER

15.

END DATA 1 BURN BACK TIME (S) = 0.20				
TIME(S) =				
VI H I				



21.

The diagram shows a control panel with two screens and a keypad. The top screen displays "END DATA 1 FILL TIME (S) = 2.00" and "TIME(S) =". Below the screen is a keypad with buttons labeled "1888", "i", "W", "9", and "i". The bottom screen displays "START MAIN END" and has five empty boxes below it. An arrow points from the top screen to the bottom screen.

END DATA 1 FILL TIME (S) = 2.00

TIME(S) =

1888 i W 9 i

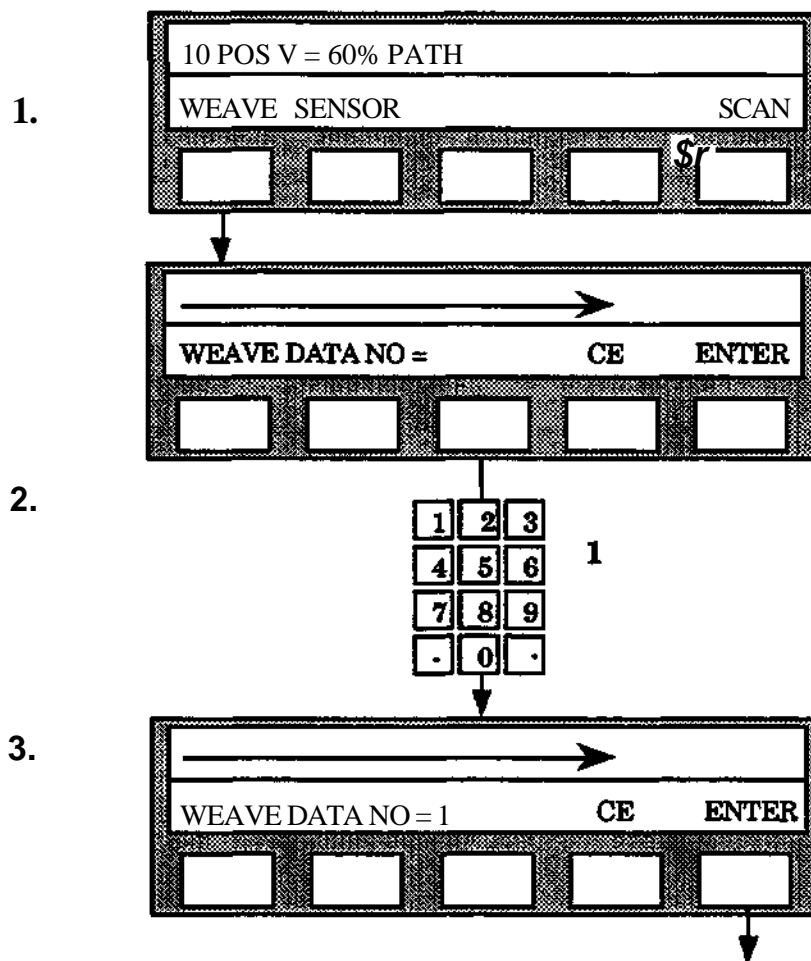
START MAIN END

Programming WEAVE DATA

The different parameters are described in Chapter 12.4.

1. Press the function button WEAVE.
2. Select required WEAVE DATA number (1-20) with the numerical button set. Not used data numbers are marked with " -" on the display.
3. Press ENTER
4. Choose some weaving pattern (ZIG-ZAG, TRIANGLE or V-SHAPED) or wrist weaving (WRIST) and then press ENTER
5. Specify required amplitude (mm) with the numerical button set.
6. Press ENTER (Now the programmed value is displayed. A new value can be entered if required).
7. Press ENTER
8. Specify required cross time (s) with the numerical button set.
9. Press ENTER (see point 6).
10. Press ENTER.
11. Specify required dwell time, Left (s), with the numerical button set.
12. Press ENTER (see point 6).
13. Press ENTER
14. Specify required dwell time, Right (s), with the numerical button set.
15. Press ENTER (see point 6).
16. Press ENTER
17. If wrist weaving is chosen, jump to point 31. Otherwise specify required dwell time, Middle (s), with the numerical button set.
18. Press ENTER (see point 6).
19. Press ENTER
20. If zig-zag weaving is chosen, jump to point 23. Otherwise specify required seam angle (deg) with the numerical button set.
21. Press ENTER (see point 6).
22. Press ENTER.
23. Specify required forward bias (mm) with the numerical button set.
24. Press ENTER (see point 6).

25. Press ENTER.
26. Specify required weave angle (deg) with the numerical button set.
27. Press ENTER (see point 6).
28. Press ENTER.
29. State whether the weaving motion always is to be perpendicular to the programmed robot path or not (YES/NO).
30. Press ENTER
31. Specify if reset is required. Then the starting menu will return.



4.

WEAVE DATA 1 TYPE ZIG-ZAG				
ZIG-ZAG	TRIANGLE	V-SHAPE	WRIST	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1

WEAVE DATA 1 TYPE WRIST				
ZIG-ZAG	TRIANGLE	V-SHAPE	WRIST	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

5.

WEAVE DATA 1 AMPLITUDE (MM) = 1.0				
AMPLITUDE =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3
Q	L	Q
Q	L	H
.	0	.

"

6.

WEAVE DATA 1 AMPUTUDEECMM) = 1.0				
AMPLITUDE = 5.5		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

7.

WEAVE DATA 1 AMPLITUDE (MM) = 5.5				
AMPLITUDE =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

8.

WEAVE DATA 1 CROSS TIME (S) = 1.00				
CROSS TIME		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

JJL3L3J

7	8	9
---	---	---

• ran

9.

WEAVE DATA 1 CROSS TIME (S) = 1.00				
CROSS TIME = 5.5		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

10.

WEAVE DATA 1 CROSS TIME (S) = 5.5				
CROSS TIME		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

11.

WEAVE DATA 1 DWELL TIME L (S) = 0.05				
DWELL TIME L =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3
4	5	6
7	8	9
-	0	.

0.55

12.

WEAVE DATA 1 DWELL TIME L (S) = 0.05				
DWELL TIME L = 0.55		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

13.

WEAVE DATA 1 DWELL TIME L (S) = 0.55				
DWELL TIME L =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

WEAVE DATA 1 DWELL TIME R (S) = 0.05				
DWELL TIME R =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

14.

1	2	3
4	5	6
7	8	9
-	0	.

0.01

15.

WEAVE DATA 1 DWELL TIME R (S) = 0.05				
DWELL TIME R = 0.01		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

16.

WEAVE DATA 1 DWELL TIME R(S) = 0.01				
• DWELL TIME R= CE ENTER				

17.

WEAVE DATA 1 DWELL TIME M (S) = 0.05				
DWELL TIME M = CE ENTER				
	11	M		

1	2	3
Q	Q	Q
7	8	9

0.01

IF WRIST
WEAVING,
PAGE 54

18.

WEAVE DATA 1 DWELL TIME M(S) = 0.05				
DWELL TIME M = CE ENTER				

19.

WEAVE DATA 1 DWELL TIME R(S) = 0.05				
DWELL TIME M = CE ENTER				

20.

WEAVE DATA 1 SEAM ANGLE (DEG) = 90				
SEAM ANGLE = CE ENTER				

1	2	3
4	5	6
7	8	9
.	0	.

80

IF ZIG-ZAG
WEAVING,
PAGE 53

21.

WEAVE DATA 1 SEAM ANGEL (DEG) = 90				
SEAMANGEL=80		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

22.

WEAVE DATA 1 SEAM ANGLE (DEG) = 80				
SEAM ANGLE =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

23.

WEAVE DATA 1 FORWARD BIAS (MM) = 0.0				
FORWARD BIAS =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3
4	5	6
7	8	9
-	0	.

1

24.

WEAVE DATA 1 FORWARD BIAS (MM) = 0.0				
FORWARD BIAS = 1		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

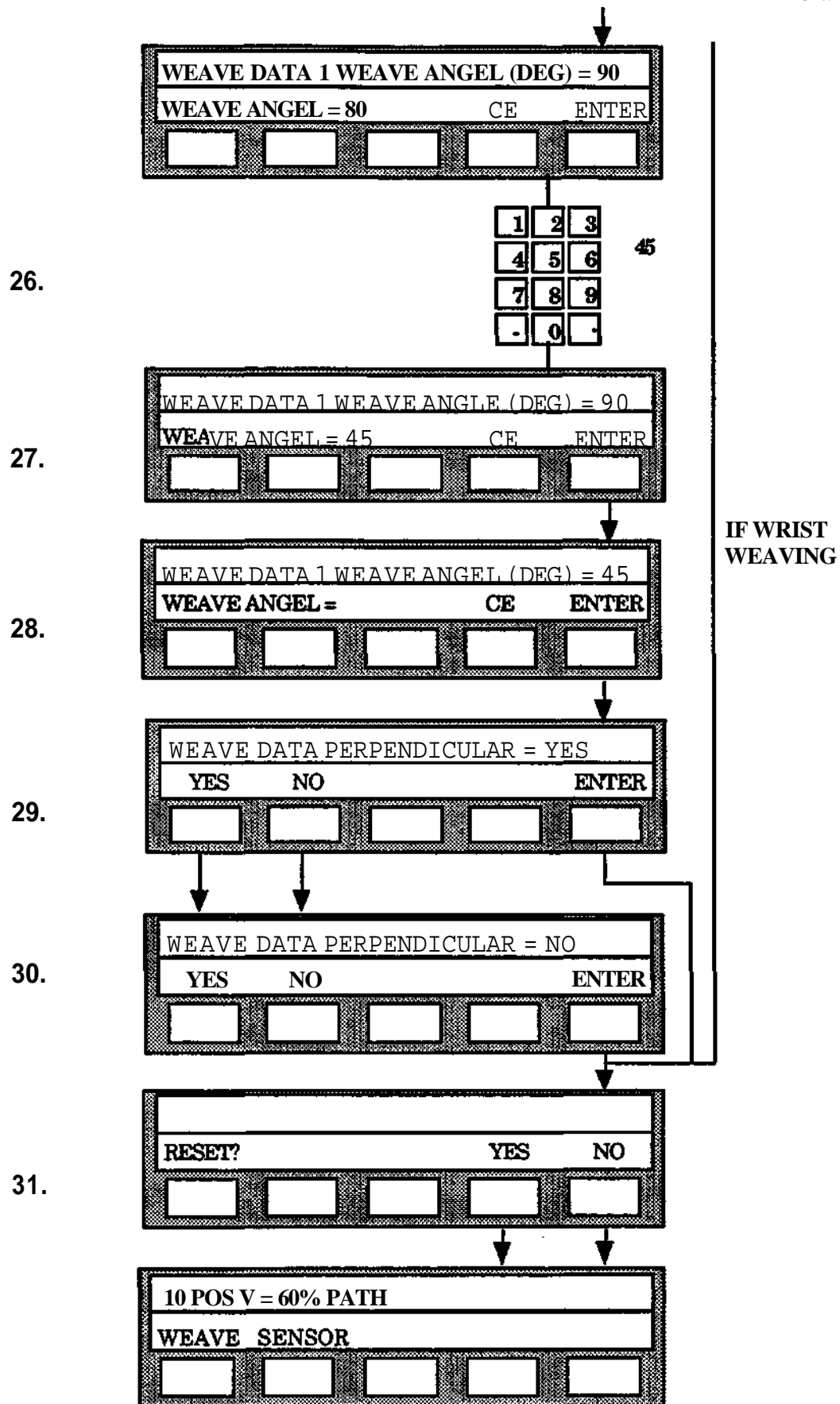
25.

WEAVE DATA 1 FORWARD BIAS (MM) = 1.0				
FORWARD BIAS =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

IF ZIG-ZAG
WEAVING

IF WRIST
WEAVING

(Continue next page)



Programming SENSOR DATA

The various parameters are described in chapter 12.4.

1. Press the function key SENSOR.
2. Select the desired SENSOR DATA -number (1-10) on the numerical keyboard. Data numbers not in use are displayed as "-".
3. Press ENTER.
4. Type the desired sensor number for side correction on the numerical keyboard.
5. Press ENTER. (The programmed value is now displayed. A new value can be entered immediately if desired).
6. Press ENTER.
7. Type the desired BIAS value on the numerical keyboard.
8. Press ENTER.
9. Press ENTER. (See point 5).
10. Type the desired sensor number for height correction on the numerical keyboard.
11. Press ENTER. (See point 5).
12. Press ENTER.
13. Type the desired sensor BIAS on the numerical keyboard.
14. Press ENTER. (See point 5).
15. Press ENTER.
16. Type the desired side correction angle on the numerical keyboard.
17. Press ENTER. (See point 5).
18. Press ENTER.

1.

POS H = 60% PATH				
WEAVE SENSOR			SCAN	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

2.

1	2	3
4	5	6
7	8	9
.	0	.

3.

SENSOR DATA NO = 1			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

4.

1	2	3
4	5	6
7	8	9
-	0	.

5.

SENSOR DATA 1 SIDE = 0 BIAS = 50				
SIDE =			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

6.

SENSOR DATA 1 SIDE = 1 BIAS = 50				
SIDE =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

SENSOR DATA 1 SIDE = 1 BIAS = 50				
BIAS =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

7.

1	2	3
4	5	6
7	8	9
.	0	.

55

8.

SENSOR DATA 1 SIDE = 1 BIAS = 50				
BIAS = 55		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9.

SENSOR DATA 1 SIDE = 1 BIAS = 55				
BIAS =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

10.

SENSOR DATA 1 HEIGHT = 0 BIAS = 50				
HIGHT =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3
4	5	6
7	8	9
.	0	.

2

11.

SENSOR DATA 1 HEIGHT = 0 BIAS = 50				
HEIGHT=2			CE	ENTER

12.

SENSOR DATA 1 HEIGHT = 2 BIAS = 50				
HEIGHT=2			CE	ENTER

13.

SENSOR DATA 1 HEIGHT = 2 BIAS = 50				
BIAS=			CE	ENTER

45		
4	5	6
7	8	9
.	0	.

14.

SENSOR DATA 1 HEIGHT = 2 BIAS = 50				
BIAS = 45			CE	ENTER

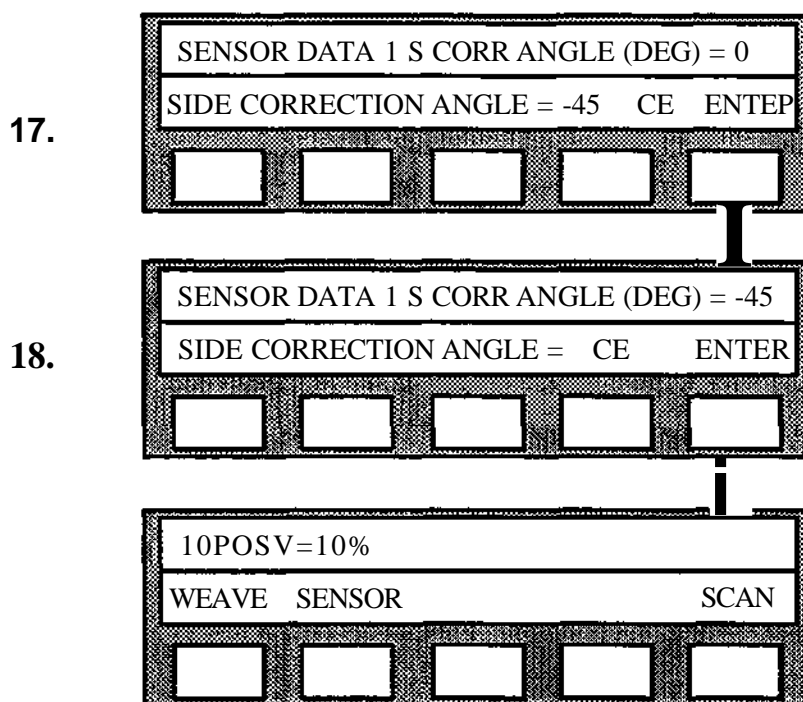
15.

SENSOR DATA 1 HEIGHT = 2 BIAS = 45				
BIAS=			CE	ENTER

16.

SENSOR DATA 1S CORR ANGLE (DEG) = 0				
SIDE CORRECTION ANGLE =			CE	ENTER

1	2	3
0	0	0
0	0	0
E	E	Q



Check **and** editing of weld data

When the programming of the weld data is complete or previously programmed weld data is to be used, it may be necessary to check and/or make amendments. This is performed simply in accordance with the following:

1. Press the control button MANUAL once and SCAN three times.
2. Press function button WDATA.
3. Select the data type required (START/MAIN/END/WEAVE/SENSOR).
4. Select the data number required.
5. Press ENTER a number of times. Each pressing steps the parameter sequence one step.
 If a value is to be changed, the new value is written in with the numerical button set and the button ENTER is pressed.

Storage of welding data

Disk

With storage of a robot program on a disk (TO DISK) the weld data is stored automatically in the same block as the robot program. The weld data area consists of the 10 START DATA, 50 MAIN DATA, 10 END DATA 20 WEAVE DATA and 10 SENSOR DATA.

The transfer of the weld data from the disk (FR DISK) can be performed in three different ways.

1. With REPLACE, the weld data in the robot is replaced with that from the disk (according to the block number selected).
2. With ADD ALL, the operator determines whether arc weld data is to be entered or not.
3. With ADD ONLY, it is possible to select the transfer of robot program (PROG) or weld data (WDATA) from the disk (according to the block number selected).

With all transfers from the disk the override register is cleared.

Computer link

In systems equipped with a computer link function, the weld data area can be stored on or accessed from the superior computer. The transfer is performed in accordance with the following:

1. Select the function TO SC or FROM SC under the control button MANUAL.
2. Select the function WDATA.
3. Specify under which block identity the weld data is to be stored.
4. The data is transferred to/from the SC.

12.5 Description of robot instructions

12.5.1 Instructions for the welding process

There are three instructions for the welding process under the control button P.

- AWELD
- WEND
- CLEAN (spatter cleaning)

These instructions are of the combination type. The first contains a positioning and call for: START DATA, MAIN DATA and WEAVE DATA and SENSOR DATA. The second contains a positioning and call of END DATA. The third contains a positioning and a call of a robot sub-program. The instructions are programmed by positioning the robot to the required point with the joystick and then pressing the P button. The robot position is then stored automatically in the memory after which one of the above instruction alternatives is selected. The rules which apply to POS instructions also apply to coordinate systems, TCPs, basic and maximum speed, positioning accuracy, editing, etc.

AWELD instruction

This instruction is used to program positioning forward to a point for welding start or weld data amendment and call for data for the weld start procedure, the weld procedure, the weaving movement of the welding gun and the sensor's method of operation. The instruction is given automatically the speed percentage which applies for the P- and POS-instruction most recently programmed and fine point. The instruction is also given the START DATA number, the MAIN DATA number, the WEAVE DATA and SENSOR DATA number which applies for the AWELD instruction most recently programmed. If required, it is possible to enter a new speed percentage, new data number and/or change the positioning accuracy to PATH or CORNER1.

The following is the procedure when the instruction is executed:

The robot is positioned to the specified location at the speed defined in the instruction. This applies if the positioning is forward to a point for weld start. If the positioning is forward to a point for weld data change (welding in progress) the weld speed specified in the instruction (included in the MAIN DATA called) applies.

When in the location specified (positioning accuracy reached) the weld start procedure is started in accordance with the START DATA given in the instruction. If SENSOR DATA are programmed the function SPS (Servo Parameter Scheduling) is disconnected to achieve constant servo log during contour following. The weld parameters VOLTAGE, CURRENT/WIRE SPEED and WELD VELOCITY (but not WEAVE DATA or SENSOR DATA) can be corrected during welding in progress with the direct action override function (see section 12.7.2). During welding, weaving is performed (if specified) and if WEAVE DATA is not 0. A correction is performed according to the data entered in the SENSOR DATA- field (sensor data *K) and ≥ 1 sensor *0). If two or more successive AWELD instructions are programmed, the first results in a weld start whereas the following only result in change of MAIN DATA and/or SENSOR DATA.

WEND instruction

This instruction is used to program positioning forward to a point for weld end and call of data for the weld end procedure. The instruction is given automatically the speed percentage (this is significant only when executing instruction by instruction, see section 12.7.4) which applies for the P- or POS-instruction most recently programmed and also fine point. The instruction is also given the END DATA number which applies for the WEND-instruction most recently programmed. If required, a new speed percentage and new data number can be entered and/or the positioning accuracy can be adjusted to PATH or COENER1.

When the instruction is executed the following takes place:

The robot is positioned to the location specified with the welding speed programmed in the MAIN DATA called by the preceding AWELD-instruction.

When in position (or at zone) the weld end procedure is executed in accordance with the END DATA specified in the instruction.

If SPS has been disconnected (contour following), it is automatically reconnected.

CLEAN instruction

Positioning of the robot to a point for spatter cleaning and call of a robot sub-program is programmed with this instruction.

The positioning is given automatically the speed percentage which applies for the P- or POS-instruction most recently programmed and PATH. The instruction is also given automatically the sub-program number which applies for the CLEAN instruction most recently programmed. If required it is also possible to enter a new speed percentage or a new subprogram number and/or to change the positioning to fine point or CORNER1.

When this instruction is executed, the subprogram is called when the positioning is completed (reached zone).

The subprogram called is to contain instructions for control of the spatter cleaning procedure.

12.5.2 Instructions for positioner

There are two instructions under the control button P for controlling external equipment such as a positioner for handling the workpiece.

- EXTC (external control)
- EXTPOS (external positioning)

These instructions are of the combination type. The first contains a positioning and call of a robot subprogram and the other contains a positioning and a transfer of position information to external equipment and acknowledgment that the transfer is concluded.

EXTC instruction

This instruction is used to program positioning of the robot to a suitable point and call of a robot subprogram. The positioning is automatically allocated the speed percentage which applies for the P- or POS-instruction most recently programmed and PATH. The instruction is also given automatically the subprogram which applies for the EXTC instruction most recently programmed. If required, a new speed percentage and a new subprogram number can be entered and/or the positioning can be changed to fine point or CORNER1.

12 Arc welding

When the instruction is executed, the subprogram is called after the positioning is complete. The subprogram called is to contain instructions to control the external equipment concerned.

EXTPOS instruction

This instruction permits positioning of external equipment without servo control.

This instruction is used to program positioning of the robot forward to a point and transmission of position information to external equipment.

The position of the external equipment is read and stored in the instruction when programming. (Position information is accessed via port 70 and transmitted via port 80. Compare with the instructions GET and TRANSFER.) The positioning is given automatically the speed percentage which applies for the P- or POS-instruction most recently programmed.

If required, a new speed percentage can be entered and/or the positioning can be changed to fine point or CORNERI.

When the instruction is executed, the position information is transferred after the positioning is completed and an acknowledgment is received.

12.6 Programming of robot instructions

General

The following functions are discussed in this section:

12.6.1 Define a position with a PATH zone

12.6.2 Instructions for the weld process

AWELD Program a weld instruction

WEND Program an end instruction

CLEAN Program a spatter cleaning instruction

12.6.3 Instructions for manipulator movements

EXTC Program an external control instruction

EXTPOS Program an external positioning instruction

12.6.4 Common functions for the instructions.

12.6.1 Define a position with a PATH zone

When the P button is pressed, the robot position is defined as a PATH. The instruction stored is identical with that obtained by pressing the POS- button.

The following function buttons are accessible after the P-button has been pressed:

AWELD WEND CLEAN EXTC

EXTPOS

There are thus three types of arc weld instructions and two types of positioner instructions available. The following sections describe in detail how these are programmed.

Note! All instruction numbers and parameter values are examples only.

12.6.2 Instructions for the weld process

Program a AWELD instruction

The AWELD function is used to program simultaneously:

- Positioning to a fine point.
- Call of START DATA and MAIN DATA and performance of a weld start.
- Call of WEAVE DATA and SENSOR DATA

When the AWELD instruction is performed, the robot will:

1. Position to the programmed position with the accuracy most recently specified.
2. a) When positioning forward to a weld start point, the weld start procedure is executed in accordance with the instruction START DATA, at the same time as weaving and welding supervision via sensors starts if required. MAIN DATA is then called for.
- b) When positioning forward to a point changing weld data, welding in accordance with the previous MAIN DATA is performed during the positioning. New MAIN DATA is then called for.

Only then is the next instruction executed. Programming of the AWELD function is described under procedure 0 below. Selection of new speed, new positioning accuracy and/or new weld data is described under the procedures A, C and D respectively in section 12.6.4.

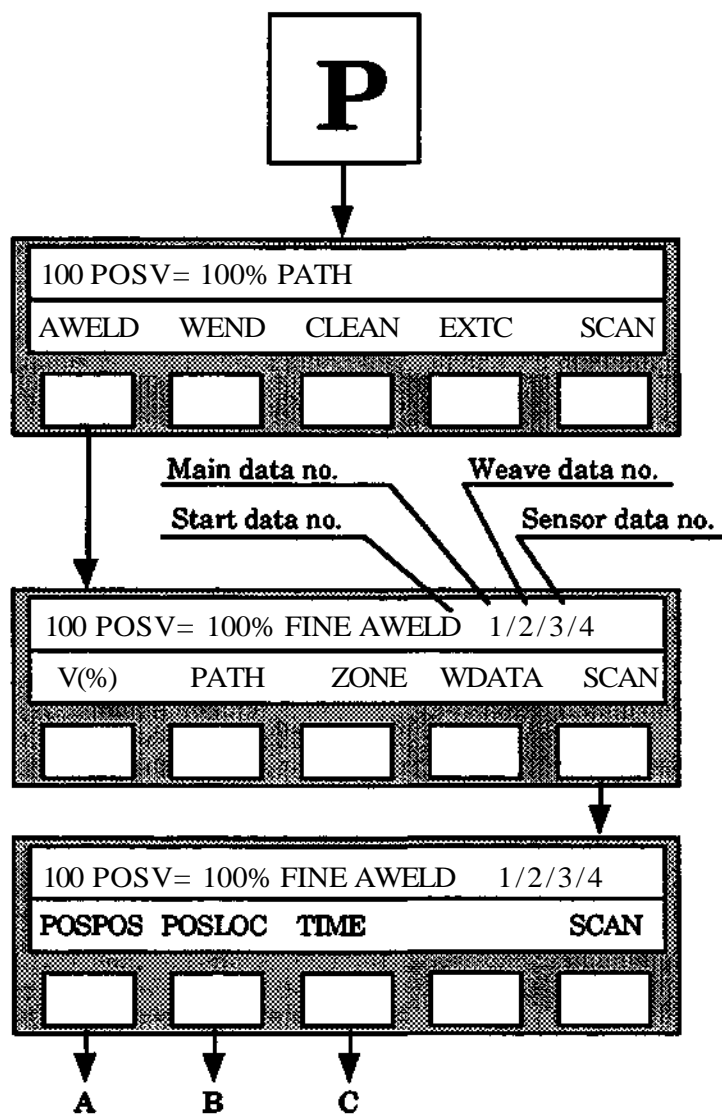
Note!

The relevant START, MAIN DATA, WEAWE and SENSOR DATA numbers are presented on the display (at the end of the instruction).

PROCEDURE 0

Program positioning with subsequent weld start and/or welding according to the following:

1. Press control button P.
2. Press function button AWELD



The AWELD instruction is now completely programmed. If a new speed, new positioning accuracy and/or new weld data are to be selected, continue with the corresponding procedures (section 12.6.4).

12 Arc welding

- A. The POS POS instruction gives the position argument for the movement in accordance with chapter 6.10.
- B. The POS LOC instruction gives the location argument for the movement, in accordance with chapter 6.12.
- C. Definition of positioning time is done with the instruction TIME as in chapter 6.15.

Note!

When the instruction TIME is given, the specified positioning time will override the given process speed.

Program a WEND instruction

The WEND instruction is used to program:

- Positioning to a fine point.
- Call of END DATA and execution of the END procedure.

When the WEND instruction is executed the robot will:

1. Position to the programmed position with the appropriate accuracy or zone most recently specified.
2. Perform a weld end in accordance with the END DATA specified in the instruction.

Only then is the next instruction executed.

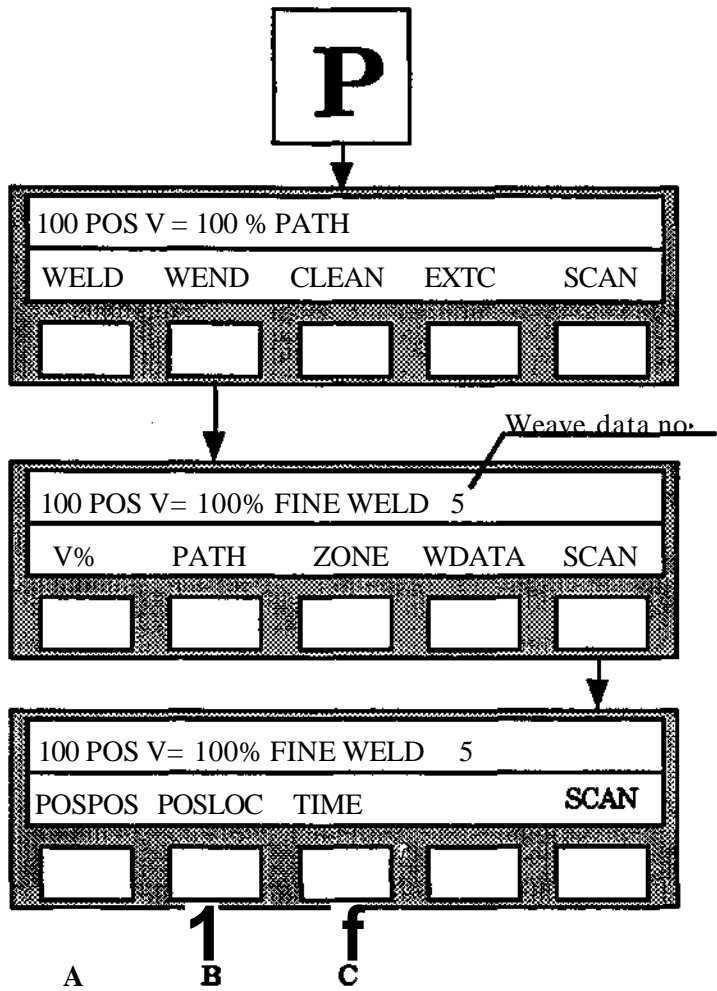
The programming of the WEND instruction is described under procedure 0 below. The selection of new speed, new positioning accuracy and/or new DATA are described under the procedures A, C and D respectively in section 12.6.4.

PROCEDURE 0

Program positioning with the subsequent weld end in accordance with the following:

1. Press control button P.
2. Press function button WEND

The WEND-instruction programming is now complete. If a new speed, new positioning accuracy and/or new weld data is to be selected, continue with the corresponding procedure (section 12.6.4).



- A. The POS POS instruction gives the position argument for the movement according to chapter 6.10.
- B. The POS LOC instruction gives the location argument for the movement according to chapter 6.12.
- C. Definition of positioning time is done with the instruction TIME as in chapter 6.15.

Note!

When the instruction TIME is given, the specified positioning time will override the given process speed.

Programming of a CLEAN instruction

The CLEAN function is used for simultaneous programming of:

- Positioning to a PATH
- Call and execution of a special subprogram for spatter cleaning

When the CLEAN instruction is executed, the robot will:

1. Position to the zone.
2. Execute the subprogram.

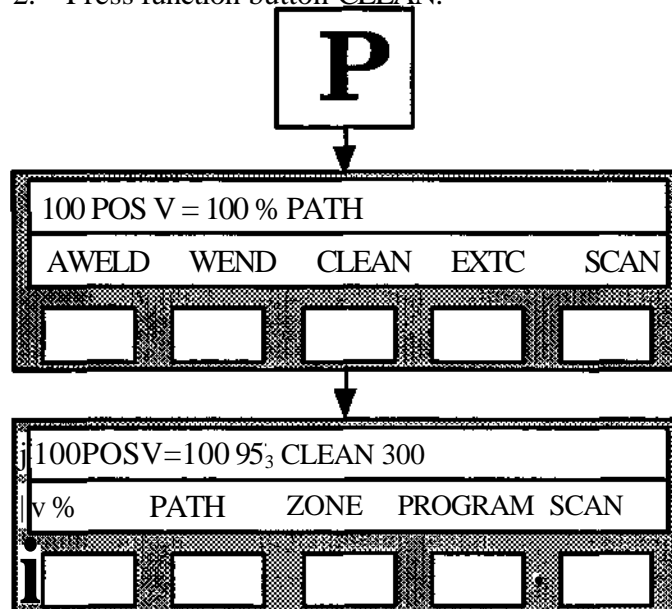
Only then is the next instruction executed. The programming of the CLEAN instruction is described under procedure 0 below.

Selection of new speed and new positioning accuracy and/or new subprogram number is described under the procedures A, C and B respectively in section 12.6.4.

PROCEDURE 0

Program positions with subsequent call of subprogram for spatter cleaning according to the following:

1. Press control button P.
2. Press function button CLEAN.



The CLEAN instruction is now programmed completely. If a new speed or a new positioning accuracy is to be selected, continue with the corresponding procedures (section 12.6.4).

12.6.3 Instructions for positioner movements

Program an EXTC instruction

The EXTC function is used to program at the same time:

- * Positioning to a fine point.
- Call and execution of a special subprogram for control of external equipment.

When the EXTC instruction is executed, the robot will:

1. Position to the programmed position at the zone currently specified.
2. Execute the subprogram.

Only then is the next instruction executed.

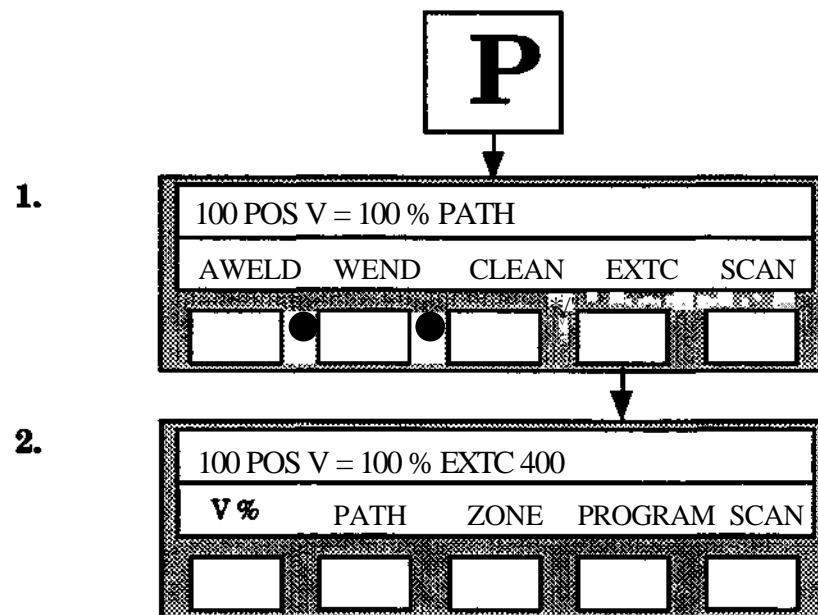
Programming of the EXTC instruction is described under procedure 0 below.

Selection of new speed, new positioning accuracy and/or new subprogram number are described under the procedures A, C and B respectively in section 12.6.4 below.

PROCEDURE 0

Program positioning with accompanying call of subprogram for external control according to the following:

1. Press start button P.
2. Press function button EXTC.



The EXTC instruction is now completely programmed. If a new speed, new positioning accuracy and/or new subprogram are to be selected, continue with the corresponding procedures (section 12.6.4).

Program an EXTPOS instruction

The EXTPOS function is used to program at the same time:

- Positioning to a PATH point
- Transfer of position information to external equipment. This position information has been accessed and stored in the instruction when programming.

When the EXTPOS instruction is executed, the robot will:

1. Position to the programmed position at the zone currently specified.
2. Transmit the position information.
3. Await acknowledgment.

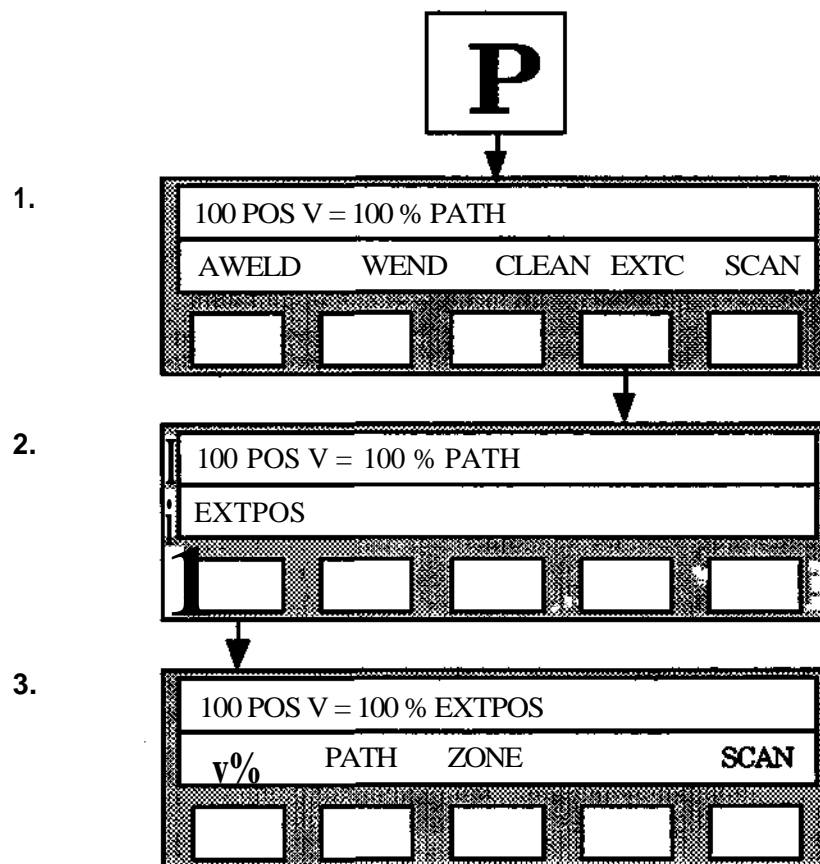
Only then is the next instruction executed. The programming of EXTPOS instructions is described under procedure 0 below.

Selection of new speed and new positioning accuracy described under the procedures A and C respectively in section 12.6.4 below.

PROCEDURE 0

Program positioning with associated transmission of position information according to the following:

1. Press control button P.
2. Press SCAN.
3. Press function button EXTPOS.



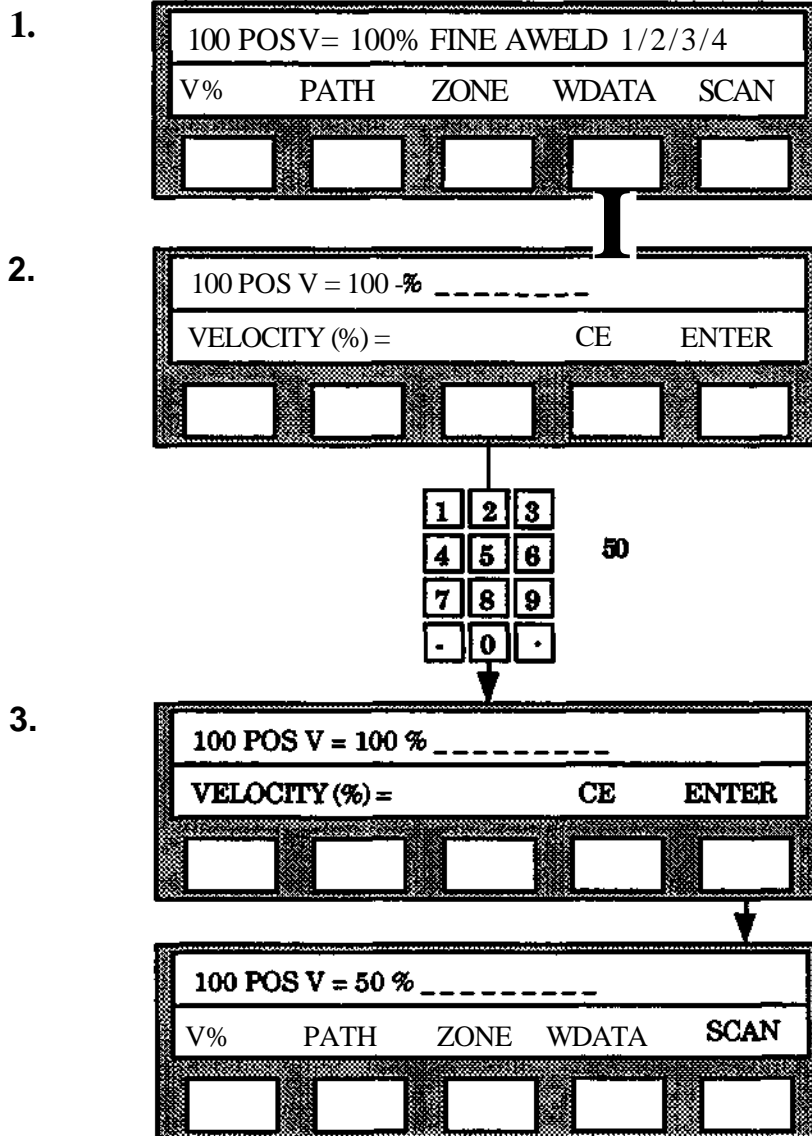
The EXTPOS instruction is now completely programmed. If a new speed and/or new positioning accuracy are to be selected, continue with the corresponding procedures (section 12.6.4).

12.6.4 Common functions

PROCEDURE A

Select a new speed percentage in accordance with the following:

1. Press function button V(%).
2. Write the speed percentage required (0.1 - 799.9 %) with the numerical button set.
3. Press function button ENTER.



PROCEDURE B

Select a new subprogram number in accordance with the following:

1. Press function button PROGRAM.
2. Write the subprogram number required (1-9999) with the numerical button set.
3. Press function button ENTER.

1.

100 POSV = 100% ----- 100				
V%	PATH	ZONE	PROGRAM	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

2.

100 POS V = 100 % ----- 100				
PROGRAM NO =			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

1	2	3	101
4	5	6	
7	8	9	
.	0	.	

3.

100 POS V = 100 % ----- 100				
PROGRAM NO = 101			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

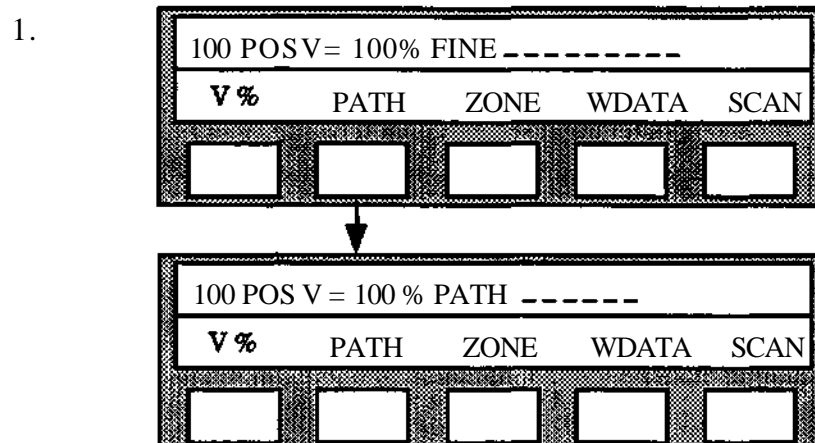
100 POS V = 100 % ----- 101				
V%	PATH	ZONE	PROGRAM	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

PROCEDURE C

Select a new positioning accuracy in accordance with the following:

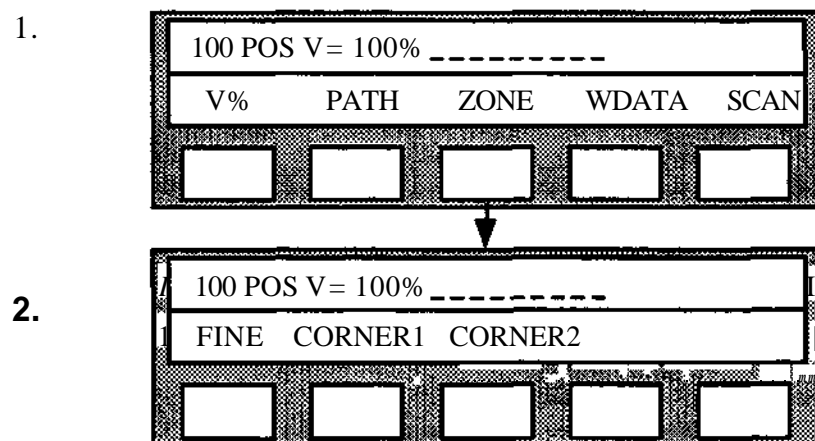
PATH POINT

1. Press function button PATH.



FINE POINT

1. Press function button ZONE.
2. Press function button FINE.



CORNER POINT

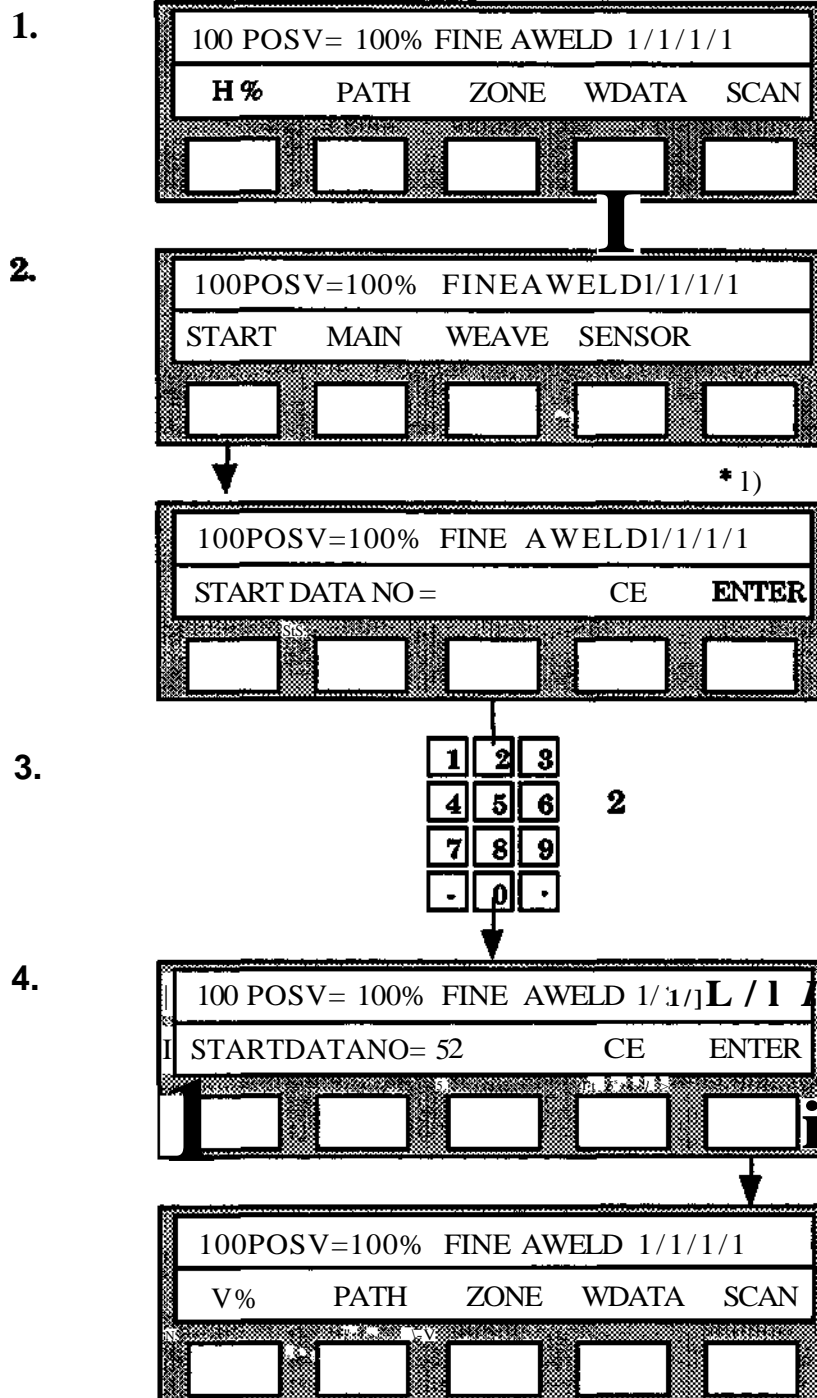
1. Press function button ZONE.
2. Press function button CORNER 1 or CORNER 2.

PROCEDURE D

Select a new data number in accordance with the following:

START, MAIN DATA, WEAVE or SENSOR DATA number

1. Press function button WDATA.
2. Press either function button START, MAIN, WEAVE or SENSOR depending on which data type is to be changed.
3. Write the new data number with the numerical button set.
4. Press ENTER



*1)The same sequence applies to MAIN and WEAVE.

END DATA number

1. Press function button WDATA.
2. Write the new data number with the numerical button set.
3. Press ENTER.

1.

100POSV=100 3h FINE WEND 1				
V%	PATH	ZONE	WDATA	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



100POSV=100% FINE WEND 1				
END DATA NO =			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

2.

1	2	3	2
4	5	6	
7	8	9	
.	0	.	



3.

100 POSV= 100% FINE WEND 1				
ENDDATANO=2			CE	ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>



100POSV=100% FINEWEND2				
V%	PATH	ZONE	WDATA	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

12.6.5 EXTFRAME

External axes are used for moving workpieces or the whole robot unit. The position and direction of these axes are not known for the robot control system and therefore interpolation between internal and external axes is not possible.

With help of the function EXTFRAME a geometric relation of an external axis and the robot coordinate system can be described. The EXTFRAME brings by that one interpolated external axis to the robot system. This means that it is possible to perform linear and circular paths with the robot and the interpolated axis at the same time. The programming is as simple as for a stillstanding external axis. Joining of tubes can, for example, be programmed with a few points even if the tube has been mounted excentric or not even parallel with a rotating interpolated axis.

The real relative velocity between the edge of the tool and the workpiece is the programmed velocity. This means that the weld parameters incl. the process velocity easily can be adjusted afterwards, without path interfering.

Programmed weaving will be carried out relative to the moving workpiece without more programming work than with a stillstanding workpiece.

Note! To avoid path following faults because of different dynamic quality of the external and internal robot axes must, in certain cases, the weld velocity be limited. To get a good result in respect of path following is it important that the KP-value for the external axis is optimized.

FUNCTION OVERVIEW

- Gear ratio for the external axis is defined under the function parameters.
- A rotating or linear external axis is defined in the EXTFRAME referring to location and direction related to the robot coordinate system.
- 20 different EXTFRAMES can be defined. 1-10 for rotating external axes, 11-20 for linear external axes.
- As standard the movement of all external axes is blocked when the EXTFRAME, for certain axis is active.
- It is possible to exclude axes, that don't influence the geometric determination of the interpolated axis, from blocking.
- Before EXTFRAME (n) is activated, external axes that have become blocked at activation, can be run to positions valid at the time of definition of the EXTFRAME (n) with the help of the function EXALIGN.
- Activation of earlier defined EXTFRAMES is done either manually, through programming or execution of the EXTFRAME instruction. EXTFRAME 0 deactivates the EXTFRAME function.

An active EXTFRAME will have the following effects on the system:

- during program execution interpolation will be carried out with the EXTFRAME active.
- all external axes except the ORBIT axes and those axes not specified as blocked axes (BLOCKAX) are blocked.

12 Arc welding

At activation several checks are conducted concerning the possibility of proper activation with respect to active REFPOINT, active STATIONS and the positions of those external axes which must have a fix position when the EXTFRAME is active (i.e. which were defined as BLOCKAXes for that specific EXTFRAME). If any of these checks fails, the system will not activate the EXTFRAME, but respond with an error message.

Before activating an EXTFRAME it is necessary to move all interfering external axes to a position where the EXTFRAME data set is valid. For this purpose it is possible to align the interfering external axes according to each defined EXTFRAME. Alignment towards an EXTFRAME means, that all external axes, which are supposed to be at a fixed position for that same EXTFRAME, are moved to the positions they had at EXTFRAME definition time. This is done under AUTO menu; EXALIGN.

To be able to work on different external axes or to handle indexed tilting of an external axis by another external device it is possible to switch between these data sets within the robot program.

EXTFRAME definition is carried out by positioning the robot's TCP to a special calibration point on the ORBIT (or on the mounted workpiece), then two times turning the axis about 90 degrees and positioning the robot's TCP to the same point on the ORBIT each time the ORBIT has been moved. From these three positions the system will calculate the movement of the external axis.

Only two points are necessary for a linear axis. There are two cases:

- When the robot stands in front of the external axis the calibration position should be fixed on the moving object (as for the rotating axis in the case above). The TCP of the robot is placed on the calibration position, there after the object is moved at least 1 meter. Finally is the robots TCP placed on the same calibration point again.
- When the robot is mounted on the trackmotion the calibration position is a position on the fix object. The robot is once moved to the calibrationpos on that object. Then the external axis (with the robot mounted) is to be moved for an optional distance (dist. should not be less than 1 mtr for accuracy reasons). Finally the robot's TCP has to be repositioned to the fix calibrationpos. Observe, that the robot don't need to have a certain relation toward the external axis it is mounted on, i.e. it is not necessary to take care of parallelity of (e.g.) extax movement direction and y-axis of the robot system.

Note that the zero (origo) position of the new coordinate system for these linear cases is in the middle, between the two calibration locations. Transport from origo can be done ± 4096 mm.

For improved accuracy a special tool with well defined TCP can be used. It is recommended not to reorientate the TCP during the calibration procedure.

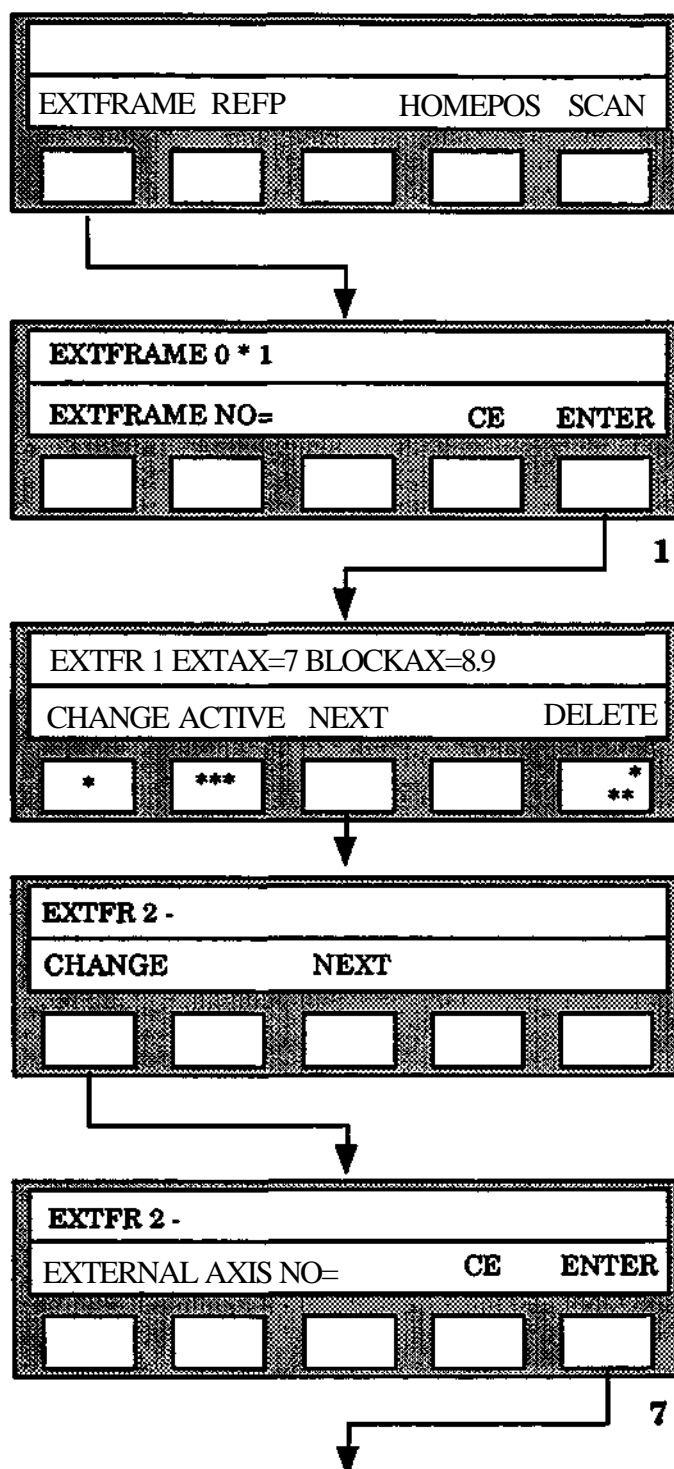
During EXTFRAME definition the movements of all other external axes, except the ORBIT axis, are blocked to avoid unwanted movements of external axes which might interfere with the calibration.

In the end of the EXTFRAME definition it is possible to define those external axes, which shall not be blocked when the EXTFRAME is active.

After the EXTFRAME definition is completed it is possible to activate the defined EXTFRAME via the pushbutton ACTIVE.

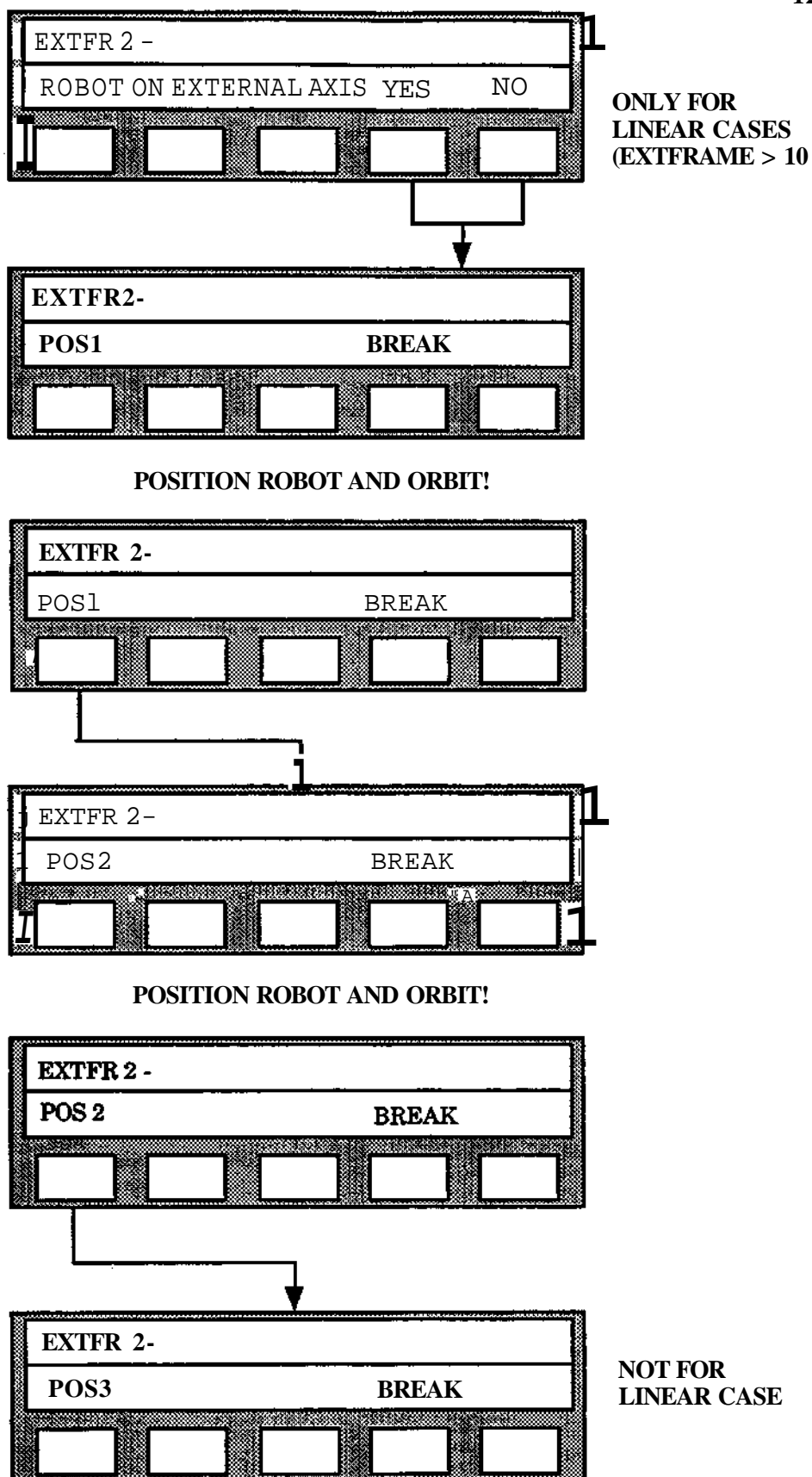


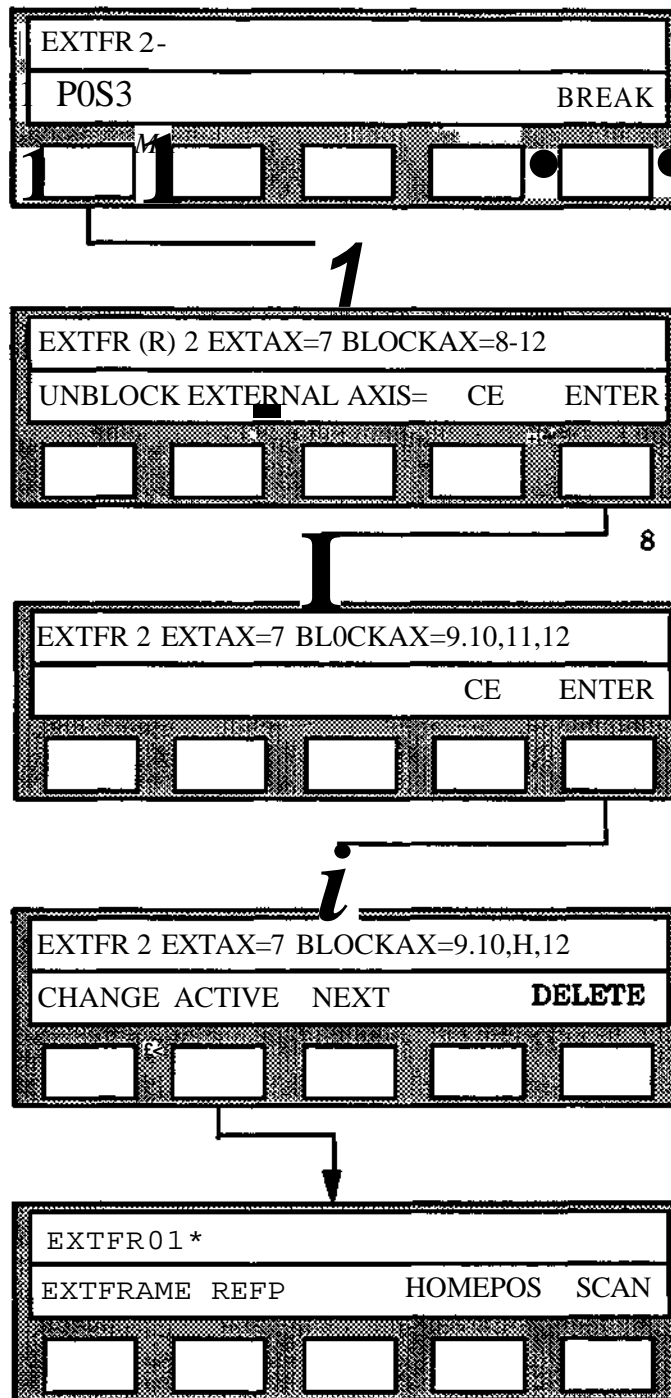
+ SCANx5



* NotforEXTFRAMEO

** Not for active or
undefined
EXTFRAME*** Not for undefined
EXTFRAME





(R) only for linear case with robot on external axis

12.6.5.1 Alignment of External axes

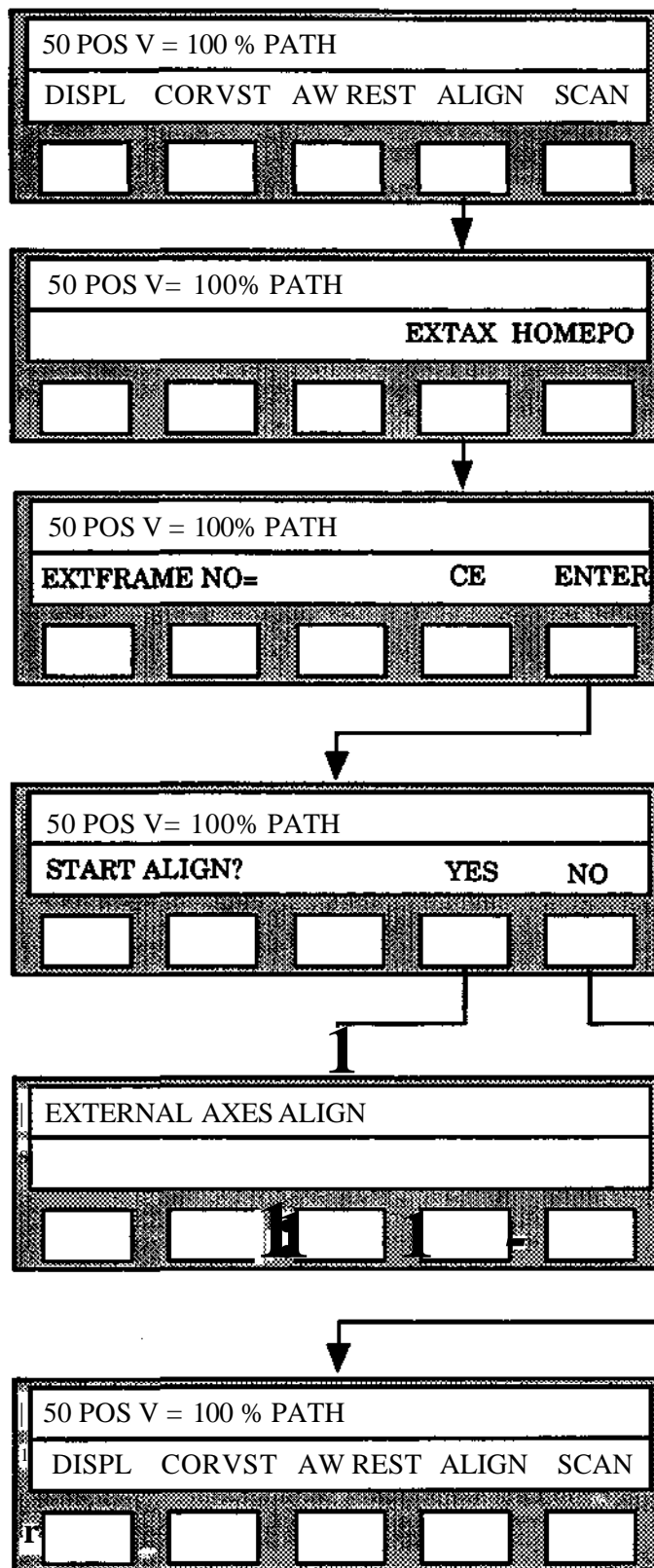
Description

One specific EXTFRAME set is valid only for a defined position of the robot relative to the external axis. There might be installations where movements of other external axes may interfere with this relation thus making the EXTFRAME data unusable. Such interfering external axes may be a bending axis of the ORBIT or a track motion of the robot.

If for some reason those "interfering" external axes have been moved before a specific EXTFRAME is to be activated, the system will react with an error message.

It will then be necessary to regain the positions of these interfering external axes to make the desired EXTFRAME data valid again. For this purpose the EXALIGN function for external axes should be used. The interfering axes (defined as BLOCKAXes in the EXTFRAME data field) will be aligned with sync-speed. No robot axis or any other external axis will be moved during alignment.

Press **AUTO + SCAN**



12.7 Program execution

12.7.1 Program test

The first version of a robot program for an arc welding application must be tested before use.

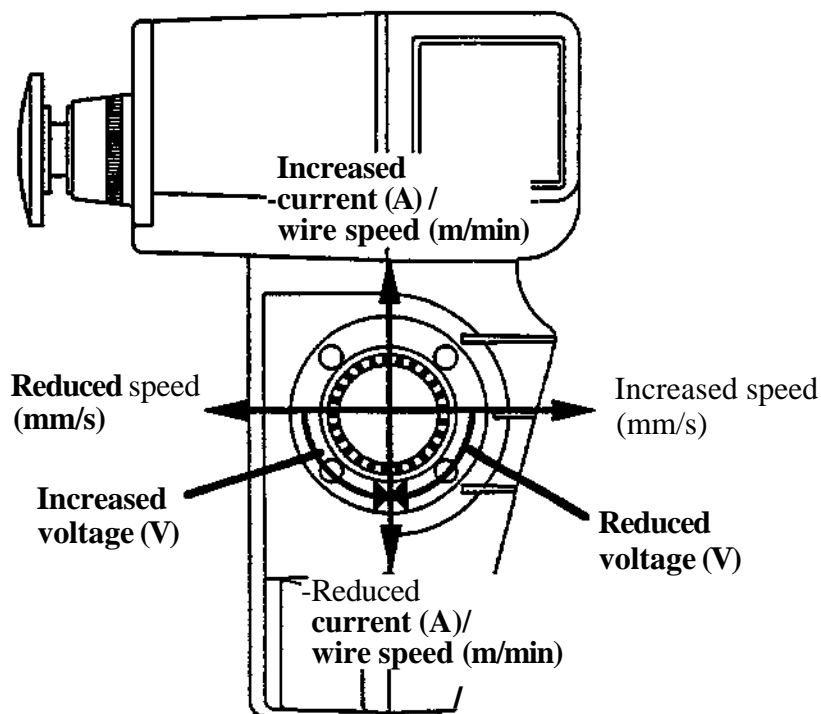
The robot system has several functions for testing and editing programs:

- **Override function** which permits individual or collective adjustment of the weld parameters CURRENT/WIRE SPEED, VOLTAGE and WELD VELOCITY during welding (see section 12.4). WEAVE and SENSOR DATA can not be adjusted in this way.
- +% and -% buttons, for adjustment of the positioning speed between the weld runs.
- **DISPLACEMENT**, which permits displacement of a certain position during program running (see programming manual).
- The possibility of blocking the welding process during program execution.
- The possibility of blocking weaving during program execution.
- The possibility of simulating input signals during program execution.

12.7.2 Override function

The robot system is provided with a direct action override function for the welding parameters speed, current/wire speed and voltage. This function is managed with the programming unit joystick which permits individual or collective adjustment of the parameters named above, during welding. The parameter values tested can be permanently stored in the program.

The three degrees of freedom of the joystick can be utilized in correcting welding data as shown in the figure below.



The change in data obtained is proportional to the deflection of the joystick. Increasing/decreasing is limited according to the table below. (Greater changes can be made by repeated use of the function.)

Parameter	Given in	Min. value	Max. value
Voltage	Volt	-40.0	40.0
Current	Amperes	-400.0	400.0
Wire speed	m/min inch/min	-20.0 -787	20.0 787
Velocity	mm/s inch/min	-40.0 -94	40.0 94

The changes of welding data obtained in this way are stored in a special register. The correction is **direct-acting**, i.e. the welding is performed with the parameters stored in the program (current MAIN DATA) **plus** the correction stored in the register. The correction can be entered permanently into the program by means of the programming unit function buttons.

Corrections can be made permanent in two different ways:

- GLOBAL
- LOCAL

Global storage means that corrected parameter values are added to the current MAIN DATA. This means that the data is changed in all weld instructions in which just this MAIN DATA is called.

Local storage means that the correction is only associated with the weld concerned (the correction is stored in the instruction). Welding with the same MAIN DATA in other parts of the program is thus not affected by local storage. Values twice the values in the table above can be stored.

These two correction storage possibilities provide almost unlimited possibilities of trimming the weld process.

The correction register mentioned above and the corrections stored locally can be cleared via the programming unit.

To maintain control over the corrections etc., the current welding data concerned can be presented during the welding in progress.

The joystick is activated for the override function by depressing a function button (ORIDE). The enabling device must be held depressed to permit use of the override function.

Note Override should be performed in MANUAL REDUCED SPEED mode. The safety function Hold-to-run can be deactivated in that mode to simplify the use of override.

12 Arc welding

The exact sequence in which buttons are pressed for the override function and welding data presentation is as follows:

1. After program start, the function button ORIDE is presented on the display. Pressing of ORIDE gives the function buttons STO.LO, CLE.LO, CLE.O, STO.GLO and presentation of the welding data concerned. In addition, two status indications are shown at the right in the display:

O = Override-register containing correction value.

L = Instruction under execution contains the correction stored.

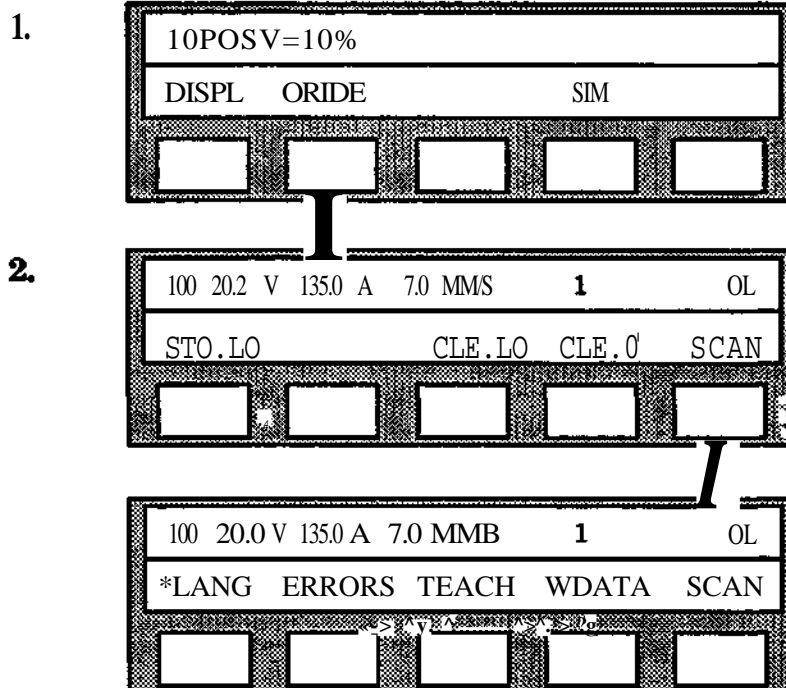
The active MAIN DATA number is also specified.

2. STO. GLO = Global storage of correction values.

STO. LO = Local storage of correction values.

CLE.LO = Clearing of correction values

CLE.O = Clearing of override.



12.7.3 Supervision of welding process

To ensure the quality of the welding performed, the robot is provided with supervision of WELD CURRENT, GAS and COOLANT. These functions are integrated in the robot control program on delivery.

The welding equipment must be provided with sensors which provide the robot with status information via reserved inputs, connected as shown in Installation S3.

Current supervision:

When the wire-feed is activated during the welding start procedure, current supervision also begins. The robot interrupts the start procedure if no indication that the weld has struck is received within the weld start max time specified in the start data. An error message is then presented on the display.

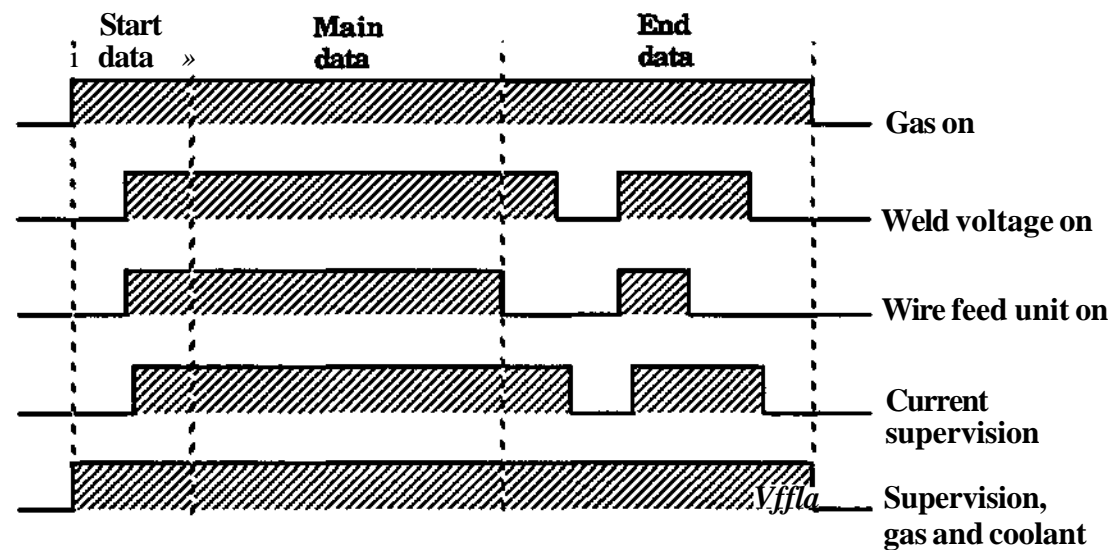
If the start sequence proceeds normally, the supervision remains during the complete welding operation and if the current indication disappears, the welding is interrupted, following a fixed sequence and an error message is presented on the display. (Very short interruptions and disturbances are not registered.)

The current supervision is deactivated when the current source is disconnected during the weld end sequence.

Gas and coolant supervision:

The supervision is initiated when the weld start procedure begins and is deactivated when the weld end procedure is concluded. If during this time the indication disappears the welding is interrupted and an error message is presented on the display.

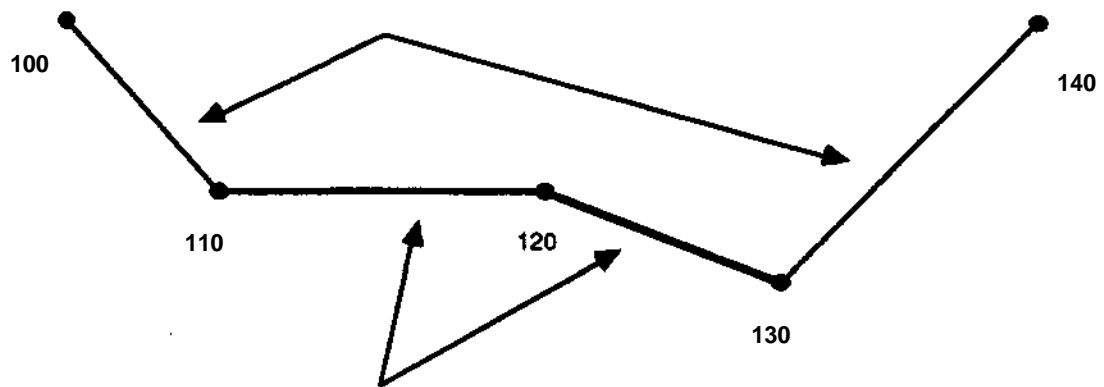
The figure below shows when the supervision is activated and concluded.



Supervision of welding process

12.7 .4 Execution of welding instructions

Special rules apply to the execution of AWELD/WEND instructions as illustrated in the following example.



100 POS V = 100% PATH
 110 POS V = 25% PATH AWELD 1/1/4/1
 120 POS V = 25% PATH AWELD 1/2/4/1
 130 POS V = 25% PATH WEND 1

- The welding is considered to start at point 110, to be changed at point 120 and is concluded at point 130.
- The robot is positioned to the point concerned and then stopped with a start instruction at the instructions 100-130. There is no welding performed with execution, instruction by instruction. The movements of the robot will be performed at the speed percentage specified in the instruction.
- With a start program at the instructions 100 or 110, the weld start will be performed at point 110. If the start program begins at instruction 120, the robot will position to this point and begin welding there.
- When executing a series of weld instructions in reverse, only the positioning will be performed, at the speed percentage specified in the instruction.
- With a start program at instruction 130, only positioning will be performed. No weld end sequence will be executed.
- If the input "Block weld process" is set, the positioning program will be executed in accordance with the above but the weld process will not be started.
- While the welding is in progress (instructions 120 and 130 above) the welding speed in the MAIN DATA is the basic speed. This has certain effects on the programming of weaving (see section 12.4).

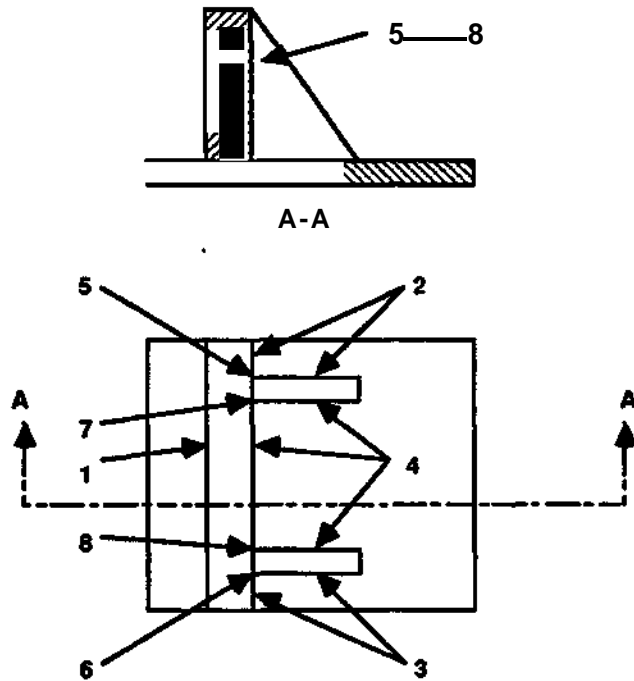
12.8 Program example

Introduction

This section describes how a robot program for arc welding can be written.

The program is assumed to be used in welding a workpiece as shown in the figure below. The different welds are numbered in the figure.

The program example is based on an installation as shown in the figure below. The positioner is assumed to be servo-powered (external axes of the robot system)



The program is to perform the following:

Set the positioner in the start position - weld 1
Reorient the object - weld 2, 3, 4, 5, 6, 7, 8.

All of the welds are started and concluded with the same START-data and END- data.
Three different sets of MAIN data are necessary:

WELD1	MAIN-data 1	WEAVE DATA 1	SENSOR DATA 4
WELD2	MAIN-data 1 and 2	WEAVE DATA 3	SENSOR DATA 2
WELD3	MAIN-data 1 and 2	WEAVE DATA 4	SENSOR DATA 4
WELD4	MAIN-data 2,1 and 2	WEAVE DATA 1	SENSOR DATA 1
WELD5-8	MAIN-data 3	WEAVE DATA 2	SENSOR DATA 1

Welding data for the welds 2, 3, 4, is exchanged during the welding.

The following program numbering has been used:

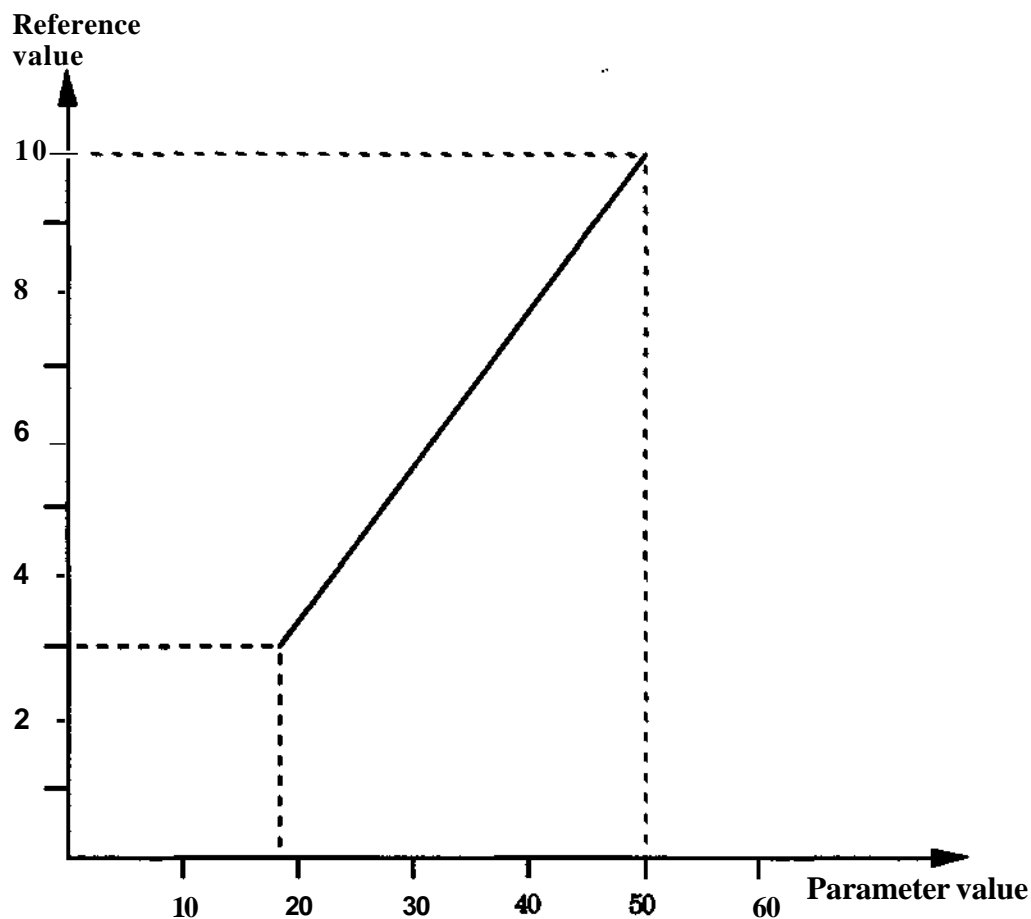
Program 0	Main program
Program 100	Weld program
Program 300	Spatter cleaning program

The following data fields numbers have been used:

START-data	No. 1
MAIN-data	No. 1 - 3
WEAVE-data	No. 1 - 3
SENSOR-data	No. 1 - 2, 4
END-data	No. 1

The working range of the power source concerned is 17 V to 50 V (parameter values selected).

The power source generates 17 V with a reference voltage of +3 V and 50 V with a reference voltage of + 10V.



In this case, the parameters are to be defined in accordance with the following:

PARAM MIN (V)	=17
PARAMMAXCV)	=50
REFMIN(V)	= 3
REF MAX (V)	=10

The max. and min. values of the parameters and reference values for current, and wire-feed unit are calculated and entered in a corresponding way.

PROGRAM 0; MAIN PROGRAM

Basic data	Speed, TCP Coordinate system
Wait for ready signal	From pushbutton on operator's panel
Call weld program	

PROGRAM 100; WELD PROGRAM

Instruction: POS	Positioning	Control of positioner to initial position
Instruction: AWELD	Pos +call of start data, main data, weave data, and sensor data.	The robot moves to the start point point for the first weld and begins welding.
Instruction: WEND	Pos+ call of end data	Conclusion of first weld.
Instruction: POS	Positioning	Control of positioning unit
Instruction: CLEAN	Pos+ call of spatter cleaning program	Positioning of robot and spatter cleaning program.
Instruction: AWELD	Pos+ call of start data, main data, weave data, and sensor data.	The robot goes forward to the start point for second weld and begins the welding.

etc.

RETURN

PROGRAM 300; SPATTER CLEANING PROGRAM

Instruction: SET OUTPUT...	Switch on compressed air	
Instruction: WATT... S	Wait	1st cleaning
Instruction: RESET OUTPUT...	Switch off compressed air	
Instruction: WATT... S	Wait	Interval between 1st and 2nd cleaning.
Instruction: SET OUTPUT...	Switch on compressed air	
Instruction: WATT... S	Wait	2nd cleaning
Instruction: RESET OUTPUT...	Switch off compressed air	

RETURN

Program list**START DATA 1**

Ignition voltage	4	V
Ignition current	-30	A
Gas pre-flow time	1	s
Hot start voltage		
Hot start current		
Hot start time	0	s

MAIN DATA 1

Weld voltage	22	V
Weld current	160	A
Weld velocity	10	mm/s

MAIN DATA 2

Weld voltage	21.5	V
Weld current	150	A
Weld velocity	8	mm/s

MAIN DATA 3

Weld voltage	21	V
Weld current	145	A
Weld velocity	6	mm/s

END DATA1

End voltage- 2	V	
End current	-15	A
Gas post-flow time	1	s
Burn-back time	0.2	s
Cooling time	0.5	s
Fill time 0.8	s	

WEAVE DATA 1-10

Type zig-zag		
Amplitude 1.0	mm	
Cross time 2	s	
Dwell time L	3	s
Dwell time R	4	s
Dwell time M	5	s
Seam angle 90	deg	
Forward bias	6	mm
Weaving angle	90	deg
Perpendicular	Yes	

SENSOR DATA 1

Side corr. sensor no.	1	
Height corr. sensor no.	2	
Side corr. sensor prestress	50	%
Height corr. sensor prestress	50	%

SENSOR DATA 2

Side corr. sensor no.	1	
Height corr. sensor no.	2	
Side corr. sensor prestress	45	%
Height corr. sensor prestress	52	%

SENSOR DATA 3

Side corr. sensor no.

1

Height corr. sensor no.

2

Side corr. sensor prestress

49

%

Height corr. sensor prestress

50

%

S3 Robot Programming Spot Welding

13	Gluing	13 Gluing
	Section	Page
13.1	Introduction	13:3
	13.1.1 General	
	13.1.2 Signals	
13.2	Programming	13:7
	13.2.1 Gluing instruction	
	13.2.2 Scaling instruction	
13.3	Overrides	13:15
	13.3.2 Override on the scaling instruction	
13.4	Error handling	13:20

13 Gluing

13.1 Introduction

13.1.1 General

The following chapter contains certain abbreviations describing the various functions and signals. Explanations of these are given below:

GLFL = GLUE FLOW
AIRFL = AIR FLOW

GLVP = GLUE FLOW SPEED VELOCITY PROPORTIONAL
AIRVP = AIR FLOW SPEED VELOCITY PROPORTIONAL

GLSP = GLUE SPEED (scaling speed for the glue flow)
AIRSP = AIR SPEED (scaling speed for the air flow)

GDELAY = GLUE DELAY
ADELAY = AIR DELAY

FLADJ = FLOW ADJUSTMENT

ORIDE = OVERRIDE

GLUE REF = GLUE REFERENCE

AIR REF = AIR REFERENCE The gluing function is a process function by means of which external gluing and sealing equipment can be controlled.

Two instructions are intended for these applications:

(1) One is a positioning instruction, which, in addition to functioning as an ordinary positioning instruction, also offers the possibility of programming GLUE- and AIR FLOWS. It is possible to select if the GLUE and AIR FLOW is to be proportional to the TCP-speed or not (the parameters GLVP and AIRVP active or inactive respectively). The following figure shows the appearance of the instructions:

9999 POSV=100% PATH					GLUEFLOW= 80% →
AIRFLOW= 0%					
GLFL	GLVP	AIRFL	AIRVP	SCAN	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

13 Gluing

(2) The second is a scaling instruction for calibrating the robot at different speeds and different gluing/sealing equipment. See the following figure:

9999 GLSP: 10V= 500 mm/s GDELAY= 0.1 s ->				
AIRSP:10V=500mm/s ADELAY= 0.1 s ->				
FLADJ= 100%				
GLSP	GDELAY	AIRSP	ADELAY	FLADJ
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Adjustment of the gluing parameters GLUE- and AIR FLOW during the gluing process is made possible with an override function (ORIDE).

An ordinary OUTPUT-instruction is used to open and close the glue gun (SET/RESET OUTPUT).

13.1.2 Signals

The gluing/sealing function uses the following input and output signals:

Analog output 1, Port 21, (0-10V):

This is the reference signal for the glue flow to the gluing equipment (GLUE FLOW).

Analog output 2, Port 22, (0-10V):

This is the reference signal for the air flow to the gluing equipment (AIR FLOW).

Digital output 7, (0/1):

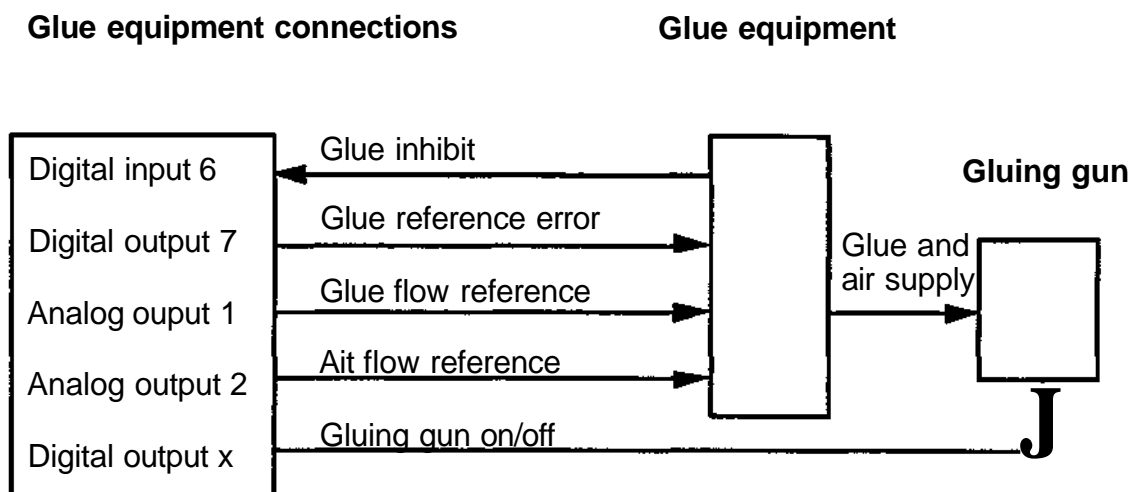
This is the signal for glue reference error. It is set to 1 when the calculated value of GLUE FLOW or AIR FLOW exceeds 10V. It is only set/reset when the parameter GLUE = 1.

Digital input 6, (0/1):

The input is used to block the gluing operation. When external equipment sets the signal to 1, the reference signals GLUE FLOW and AIR FLOW are reset to zero as soon as the current execution of an instruction is concluded.

In addition, a digital output must be used to open and close the glue gun. This output and the analog outputs 1 and 2 also have a delaying function which permits compensation, during the execution of the program, for lag in the robot movement and for delay in the gluing equipment.

The following figure shows the connections described:



NOTE! Digital input 6 and digital output 7 are dedicated for process control. The user is to ensure that the input and the output are not used for other instructions.

If an error situation occurs, it is also possible for external equipment to close off the GLUE FLOW and the AIR FLOW by setting digital input 6 to 1. The input then resets the reference signals as soon as the current instruction execution is completed and before execution of the next instruction. This input is mainly intended for testing of robot programs. The analog outputs are however limited to 10V. When any of the reference signals exceed 10V, the calculation are changed somewhat. See the following section.

10.1.1.3 Calculation of the reference signals

(1) If GLUE FLOW or AIR FLOW are proportional to the TCP speed (GLVP or AIRVP are active), the calculation is performed in accordance with the following general equation:

$$\text{new GLUE/AIR-REF} := \text{FLADJ} * \text{FLOW} * \text{TCPSP} / \text{SCALESP} * 10\text{V}$$

where Flow Adjustment (FLADJ) is a master override function which affects all flow instructions. This and the GLUE/AIR flow (FLOW) are given as percentages. The TCP speed (TCPSP) and the scaling speed for the GLUE/AIR flow (SCALESP) programmed, are normally given in mm/s or inch/min. 10V is a correction factor.

(2) If however the GLUE FLOW or the AIR FLOW are not proportional to the TCP speed (GLVP or AIRVP is inactive), the calculation is performed in accordance with the following equation:

$$\text{new GLUE/AIR-REF} := \text{FLADJ} * \text{FLOW} * 10\text{V}$$

Note that all of the factors affect the new output reference. Even if the TCP speed and the GLUE/AIR speed are equal, the new reference need not be 10V. This also depends on any flow adjustment and the GLUE/AIR flow.

Example 1:

The glue reference is thus calculated in accordance with the following:

$$\text{GLUE REF} := \text{FLADJ} * \text{GLFL} * \text{TCPSP} / \text{GLSP} * 10\text{V}$$

Programmed FLADJ : 80%
 Programmed GLFL : 90%
 Programmed GLSP : 10V = 500 mm/s
 Calculated TCPSP : 400 mm/s

If GLVP is active, then $\text{GLUE REF} := 0,8 * 0,9 * 400 / 500 * 10\text{V} = 5,76\text{V}$

Otherwise $\text{GLUE REF} := 0,8 * 0,9 * 10\text{V} = 7,2\text{V}$

Example 2:

The air reference is calculated in a corresponding way in accordance with the following:

$$\text{AIR REF} := \text{FLADJ} * \text{AIRFL} * \text{TCPSP} / \text{AIRSP} * 10\text{V}$$

Programmed FLADJ : 120%
 Programmed AIRFL : 75%
 Programmed AIRSP : 10V = 650 mm/s
 Calculated TCPSP : 400 mm/s

If AIRVP is active, then $\text{AIR REF} := 1,2 * 0,75 * 400 / 650 * 10\text{V} = 5,54\text{V}$

Otherwise $\text{AIR REF} := 1,2 * 0,75 * 10\text{V} = 9,0\text{V}$

The analog outputs are however limited to 10V. There is an exception from the normal calculations when one of the reference signals (GLUE/AIR REF) exceeds 10V.

When both signals are proportional to the TCP speed, the higher value is limited to 10V and the lower is reduced proportionally to maintain the relation between the two reference signals (see the formula above). This applies even when both signals are not proportional to the TCP speed, i.e. when GLVP and AIRVP not active.

If only one of the signals is proportional to the TCP speed, the values are only limited to a maximum of 10V without retaining the relation.

Example:

Calculated GLUE REF : 15V
 Calculated AIR REF : 12V

(a) Both signals proportional to the TCP speed.

As $\text{GLUE REF} > \text{AIR REF}$, the outputs are set to:

GLUE REF := 10V
 AIRREF := $12/15 * 10\text{V} = 8\text{V}$

(b) Only GLUE REF is proportional to the TCP speed.

The outputs are then set to:

GLUE REF := 10V
 AIRREF := 10V

Digital output 7 is set to 1 (Digital output 7 := 1) at the same time and remains so while the reference signals are limited.

13.2 Programming

13.2.1 Gluing instruction

The gluing instruction functions as a normal positioning instruction with the added capability of setting reference signals for the glue and air flows. When a gluing instruction is executed, the robot positions to the zone of the position programmed and sets the glue and air signals to the voltages which correspond to the percentages for GLUE- and AIR FLOW. These are retained until a new gluing instruction is performed.

Note that the two percentages programmed in the gluing instruction apply to different robot movements.

The percentage for the speed ($V=x\%$) applies to the robot track **before** the programmed position, i.e. the speed at which the robot positions to the point programmed.

The percentages for the glue/air flows ($GLFL=x\%$ and $AIRFL=x\%$) apply to the the robot track after the programmed position, i.e. the magnitude of the glue flow when the robot leaves the point concerned. See the following figure:



With respect to the coordinate system, TCP, basic and maximum speed, positioning accuracy, editing etc. the same rules apply as for normal positioning instructions.

The instruction is programmed by positioning the robot to the point required and then pressing the P-button. The menu for the instruction parameters appears as follows:

9999 POSV=100% PATH GLUEFLOW= 80% ->				
AIRFLOW= 0%				
GLFL	GLVP	AIRFL	AIRVP	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Four different position types are available: PATH, C1, C2, FINE.

GLFL (Glue Flow), Range 0-100%, Resolution 1%:

Analog output 1 is set to a value corresponding to the percentage specified, 10V corresponds to 100%. Remember that the glue flow signal is also affected by the global override parameter FLADJ (Flow Adjustment), and by the TCP speed if the parameter GLVP is active.

AIRFL (Air Flow):

As for GLFL except for air flow and for analog output 2.

13 Gluing

GLVP (Glue Flow Speed Velocity Proportional), Value: activeAnactive:

If the parameter is active, the glue flow is scaled proportional to the TCP speed. An activated GLVP parameter is indicated with the letter (V) after the glue flow in the programming menu. The parameter is activated/deactivated by pressing the button GLVP as shown in the example below.

ABRVP (Air Flow Speed Velocity Proportional):

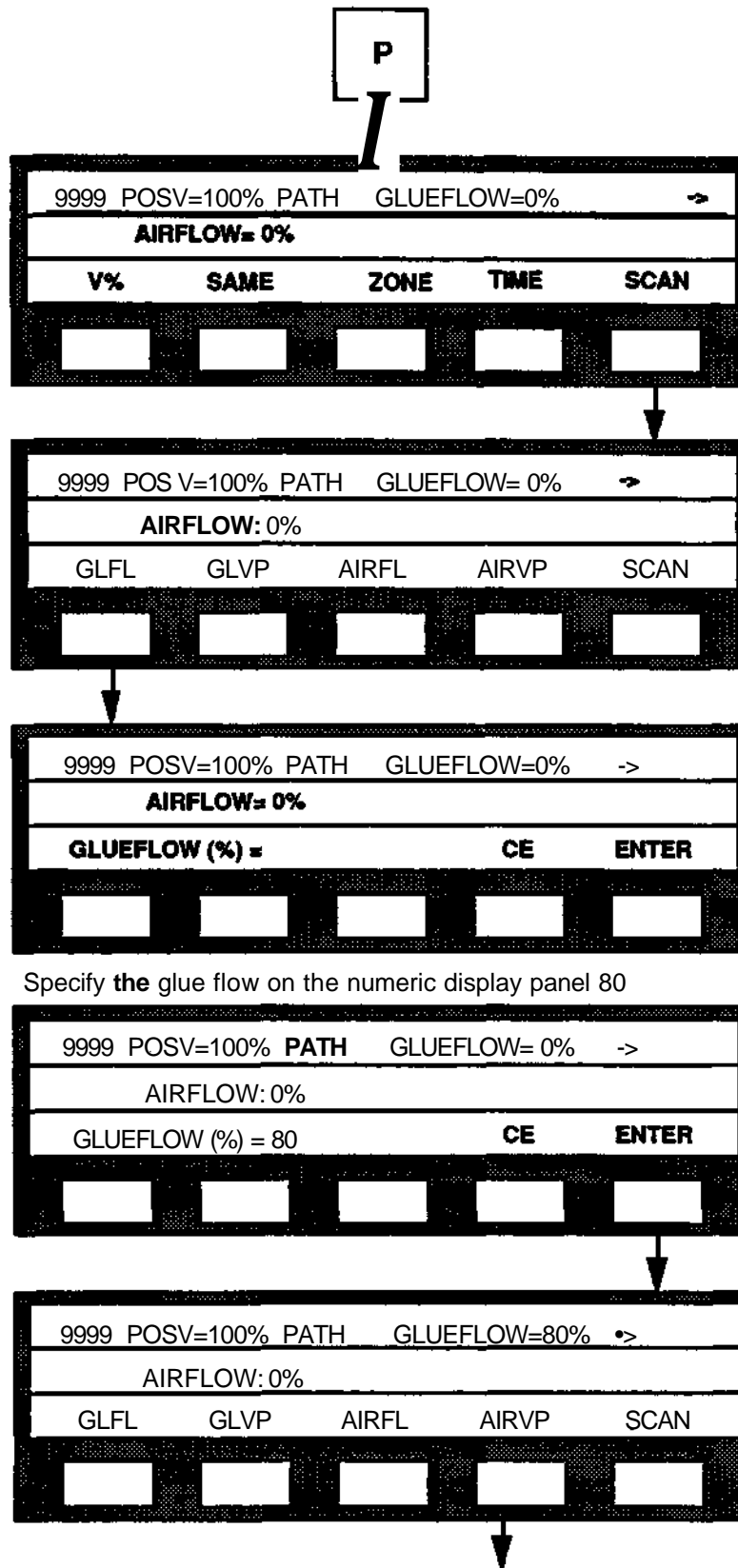
As for GLVP except for air flow.

Default values:

POSTYPE	:PATH
GLFL	:0%
AIRFL	:0%
GLVP	: inactive
AIRVP	: inactive

Example:

Gluing instructions are programmed via the P-button. See the following figure:



Specify the glue flow on the numeric display panel 80

9999 POSV=100% PATH GLUEFLOW= 80% ->				
AIRFLOW= 0%				
AIRFLOW (%) =		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Specify the air flow on the panel

50

9999 POSv=100% PATH GLUEFLOWs 80% ->				
AIRFLOW= 0%				
AIRFLOW (%) = 50		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 POSV=100% PATH GLUEFLOWs 80% ->				
AIRFLOW= 50%				
GLFL	GLVP	AIRFL	AIRVP	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 POSV=100% PATH GLUEFLOW= 80% (V) ->				
AIRFLOW= 50%				
GLFL	GLVP	AIRFL	AIRVP	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

The parameter is deactivated
and (V) disappears when
GLVP is pressed once again:

9999 POSV=100% PATH GLUEFLOW=80% ->				
AIRFLOW= 50%				
GLFL	GLVP	AIRFL	AIRVP	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 POSV=100% PATH GLUEFLOWs 80% ->				
AIRFLOW= 50% (V)				
GLFL	GLVP	AIRFL	AIRVP	SCAN
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

13.2.2 Scaling instruction

The scaling instruction enables the user to program the relation between the calculated TCP speed and the output voltage on the two analog channels. The instruction also includes a global override (FLADJ) which functions as a type of scale factor in all subsequent GLUE/AIR instructions until a new scaling instruction is executed.

The scaling instruction is normally located at the beginning of the program, as nominal speed, reference frame etc. However, the instruction can be used anywhere in the program if the scaling values are to be changed. If no scaling instruction is given, the preset values apply as default values. The menu for the instruction parameters has the following appearance:

9999 GLSP: 10V= 500 mm/s GDELAYs 0.1 S->				
AIRSP:10V=500mm/s ADELAY= 0.1 s ->				
FLADJ= 100%				
GLSP	GDELAY	AIRSP	ADELAY	FLADJ
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

GLSP (Glue Speed):

If the robot controls the gluing equipment by using the TCP speed as a reference, a reference speed must be defined for the maximum voltage 10V at the analog output. This is done by setting a value for the glue speed (mm/s or inch/min) which corresponds to 10V.

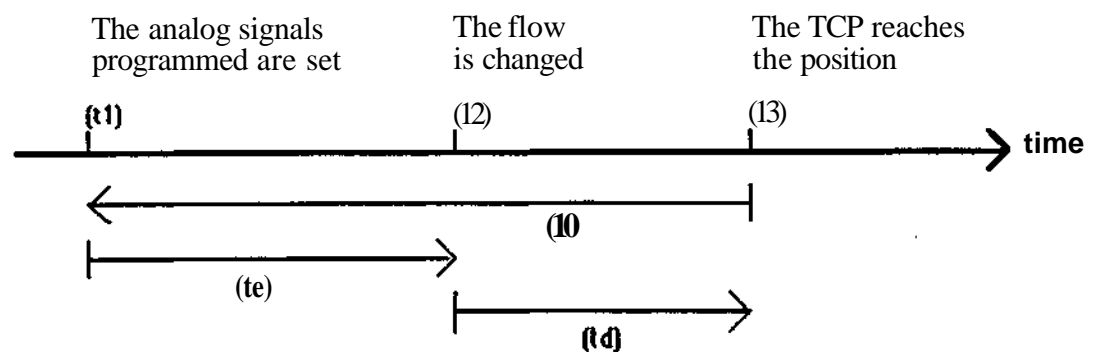
AIRSP (Air Speed):

As for GLSP except for air.

GDELAY (Glue Delay):

The updated analog output signal is set a certain time before the TCP reaches the position where the new value of the output signal is programmed. The signal thus arrives before the TCP. It is therefore possible to delay the new reference signal to permit compensation during program execution.

The robot reaches the position concerned a certain time, depending on the TCP speed, after the analog signal is set. In addition, the lag varies in different gluing equipment and the delay must be adjusted in each particular case. The following figure shows the delay times:



13 Gluing

The analog reference signals GLUE FLOW and AIR FLOW are set at the same time (t1). This occurs the time (tf) before the time at which the TCP reaches the position. The gluing equipment changes the flow the time (te) after the change in the reference signals if no extra delay is introduced. To force the change in flow to the time (t3) when the TCP has reached the position, the delay time (td) must be programmed in the scaling instruction. If the delay is set to zero, the flow will change somewhat before the TCP reaches the position programmed (provided that the gluing equipment is faster than the robot).

ADELAY (Air Delay):

As for GDELAY except for air.

FLADJ (Flow Adjust):

The flow adjustment factor functions as a general override in all of the subsequent flow instructions until the next scaling instruction is reached or program execution concludes. This applies to both GLUE FLOW and AIR FLOW.

Argument range:

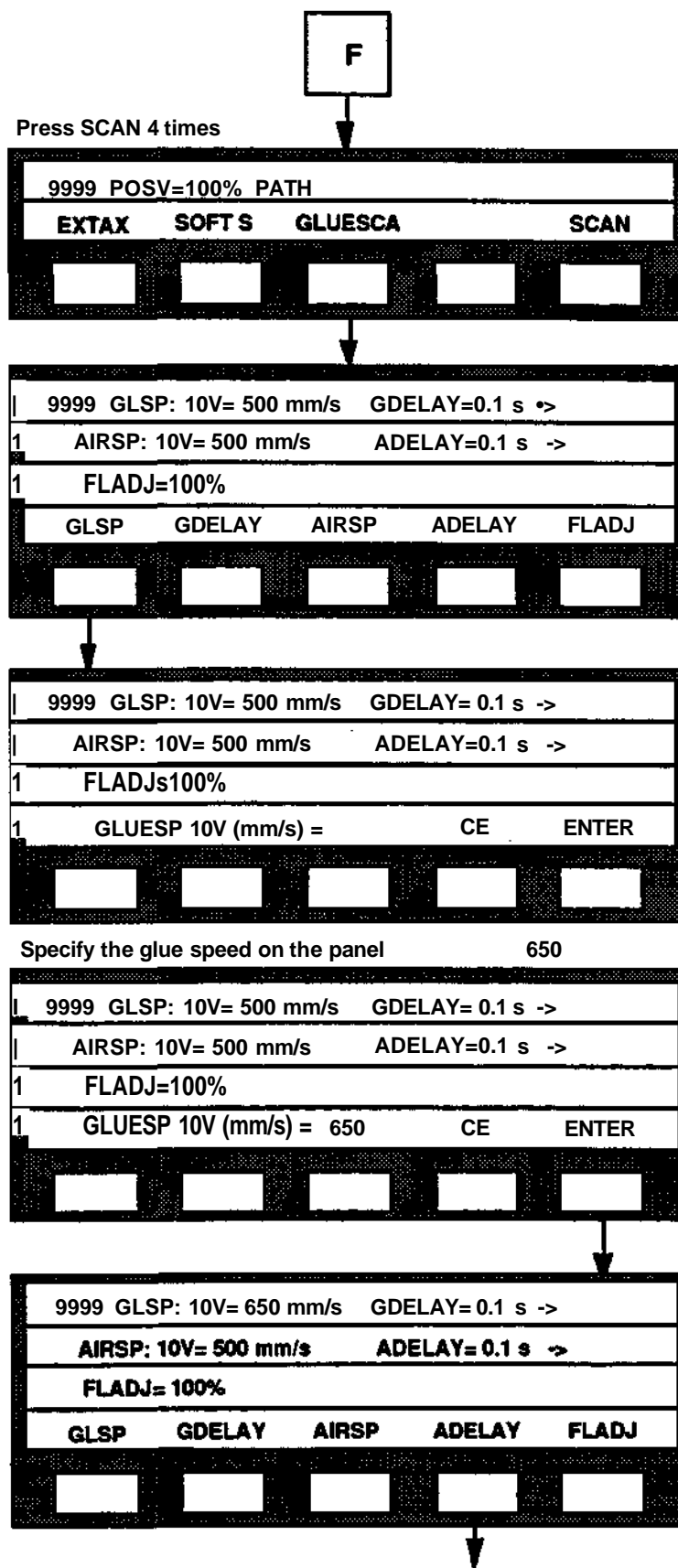
GLSP : 100 - 2000 mm/s	resolution 1 mm/s
(236 - 4724 inch/min)	resolution 1 inch/min)
AIRSP: 100 - 2000 mm/s	resolution 1 mm/s
(236 - 4724 inch/min)	resolution 1 inch/min)
GDELAY : 0 - 0,5 s	resolution 0,01 s
ADELAY : 0 - 0,5 s	resolution 0,01 s
FLADJ : 50 - 200%	resolution 1%

Default values:

GLSP : 500 mm/s
AIRSP: 500 mm/s
GDELAY : 0,1 s
ADELAY : 0,1 s
FLADJ : 100%

13 Gluing

The scaling is located under the F-button. See the following figure:



1	9999 GLSP: 10V= 650 mm/s	GDELAY= 0.1 s ->
	AIRSP: 10V= 500 mm/s	ADELAY= 0.1 s ->
1	FLADJs 100%	
	AIRDELAY (s) =	CE ENTER
	<input type="text"/>	<input type="text"/>

Specify the air delay on the panel

0.25

	9999 GLSP: 10V= 650 mm/s	GDELAY= 0.1 s ->
1	AIRSP: 10V= 500 mm/s	ADELAY= 0.1 s ->
	FLADJ=100%	
1	AIRDELAY (s) = 0.25	CE ENTER
	<input type="text"/>	<input type="text"/>

	9999 GLSP: 10V= 650 mm/s	GDELAYs 0.1 S ->			
	AIRSP:10V= 500 mm/s	ADELAY= 0.25 s ->			
	FLADJ= 100%				
	GLSP	GDELAY	AIRSP	ADELAY	FLADJ
	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

	9999 GLSP: 10Vr 650 mm/s	GDELAY= 0.1 s ->
	AIRSP: 10V= 500 mm/s	ADELAY= 0.25 s ->
	FLADJ= 100%	
	FLOW ADJUST (%) =	CE ENTER
	<input type="text"/>	<input type="text"/>

Specify the flow adjustment on the panel

50

	9999 GLSP: 10V= 650 mm/s	GDELAY= 0.1 s ->
	AIRSP: 10V= 500 mm/s	ADELAY= 0.25 s ->
	FLADJ= 100%	
	FLOW ADJUST (%) = 50	CE ENTER
	<input type="text"/>	<input type="text"/>

13.3 Overrides

It is possible, with the override function ORIDE, to control the GLUE/AIR flow for individual gluing instructions (local override), or via the scaling instruction FLADJ (global override).

Changing the reference signals with ORIDE during automatic program execution affects the reference signals the next time a gluing instruction is executed.

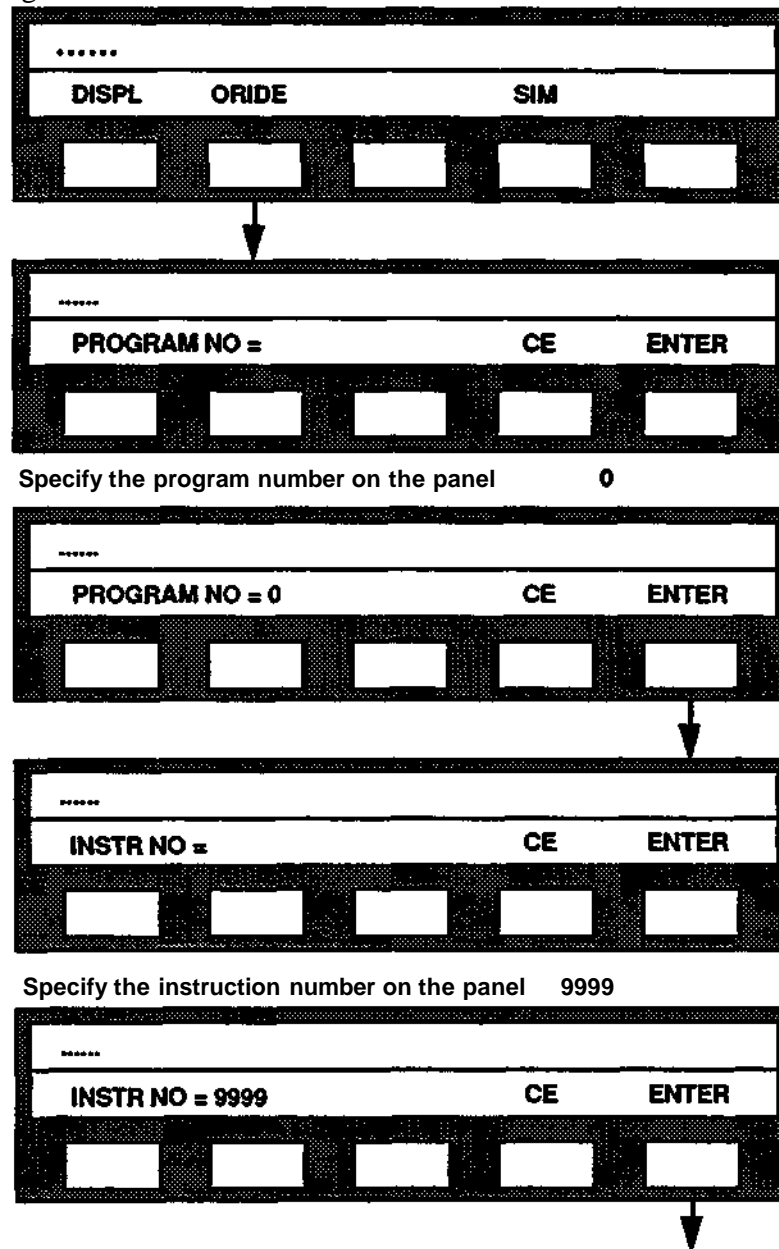
There are thus two principally different overrides:

- (1) One which affects the GLUE/AIR flow.
- (2) One which affects the scaling instruction FLADJ.

The instruction number is requested when the ORIDE button is pressed during program execution. The instruction number specified determines which type of override function is applied. If the instruction number is that of a GLUE/AIR function, it becomes a local override, but if the number is that of a scaling instruction, it becomes a global override.

13.3.1 Override on the GLUE/AIR instruction

The following figure shows how this ORIDE function is used:



9999 POSV=100% PATH GLUEFLOWr 80% (V)->				
AIRFLOW= 75% (V)				
GLUEFLOW (%) =		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Specify the glue flow on the panel 100

9999 POSV=100% PATH GLUEFLOWs 80% (V) ->				
AIRFLOW= 75% (V)				
GLUEFLOW (%) = 100		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 POSV=100% PATH GLUEFLOW= 100%(V)->				
AIRFLOW= 75% (V)				
AIRFLOW (%) =		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Specify the air flow on the panel 60

9999 POSV=100% PATH GLUEFLOWr 100% (V)->				
AIRFLOW= 75% (V)				
AIRFLOW (%) = 60		CE		ENTER
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 POSV=100% PATH GLUEFLOWr 100% (V)->				
AIRFLOW= 60% (V)				
DISPL	ORIDE	NEXT	BWD	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

It is possible to continue in the program with the NEXT or BWD buttons. When an instruction is affected which is neither a scaling instruction nor a GLUE/AIR instruction, the system will not go to the submenu for entry of new values for Flow Adjustment (FLADJ) or GLUE/AIR flow. The robot then shows only the current instruction on the top line of the display of the programming unit. It is however possible to step farther through the program by pressing the NEXT button until a scaling instruction or a GLUE/AIR instruction is encountered.

13.3.2 Override on the scaling instruction

This override function is used in a way similar to that described above. See the following figure:

The figure illustrates the process of overriding a scaling instruction through a series of five control panel screens:

- Panel 1:** Shows the initial state with fields for **DISPL**, **ORIDE**, and **SIM**. Below these are five input boxes. An arrow points down to the next panel.
- Panel 2:** The **ORIDE** field is active, showing **PROGRAM NO =**. Below it are five input boxes. The first box contains the digit **0**. An arrow points down to the next panel.
- Panel 3:** The **ORIDE** field now displays **PROGRAM NO = 0**. Below it are five input boxes. An arrow points down to the next panel.
- Panel 4:** The **ORIDE** field is active, showing **INSTR NO =**. Below it are five input boxes. The first box contains the digit **9**, the second **9**, the third **9**, and the fourth **9**. An arrow points down to the next panel.
- Panel 5:** The **ORIDE** field now displays **INSTR NO = 9999**. Below it are five input boxes. An arrow points down from the last box.

Text labels between panels indicate the input values: "Specify the program number on the panel 0" and "Specify the instruction number on the panel 9999".

9999 GLSP: 10V= 650 mm/s GDELAY= 0.1 s ->				
AIRSP: 10V= 500 mm/s ADELAY::0.25 s ->				
FLADJ=100%				
FLOW ADJUST (%) =		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Specify the flow adjust on the panel 120

9999 GLSP: 10V= 650 mm/s GDELAY= 0.1 s ->				
AIRSP: 10V= 500 mm/s ADELAYr 0.25 s ->				
FLADJ=100%				
FLOW ADJUST (%) = 120		CE	ENTER	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

9999 GLSP: 10V= 650 mm/s GDELAY= 0.1 s ->				
AIRSP: 10V= 500 mm/s ADELAY= 0.25 s ->				
FLADJ=120%				
DISPL	ORIDE	NEXT	BWD	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

It is also possible here to continue in the program with the NEXT or BWD buttons. When an instruction is affected which is neither a scaling instruction nor a GLUE/AIR instruction, the robot will not go to the submenu for entry of new values for Flow Adjustment (FLADJ) or GLUE/AIR flow. The robot then shows only the current instruction on the top line of the display of the programming unit. It is however possible to step further through the program by pressing the NEXT button until a scaling instruction or a GLUE/AIR instruction is encountered.

13.4 Error handling

As described in section 13.1.2, the calculation of reference signals can result in values which exceed 10V, the maximum permitted value. In such a case, the higher value is limited to 10V and the lower is scaled down proportionally to maintain the relation between the signals GLUE REF and AIR REF.

At the same time, digital output 7 is set to 1 while the references are limited. Program execution continues however.

The reference signals are reset to zero when execution of the program stops or with an emergency stop (GLUE REF and AIR REF := OV).

14 Spot welding

Section	Page
14.1 Introduction	14.3
14.2 The standard function	14.3
14.2.1 General	
14.2.2 Welding gun, manual operation	
14.2.3 Function parameters	
14.2.4 The spot weld instruction	
14.2.5 Programming the first spot weld instruction	
14.2.6 Subsequent instructions	
14.2.7 Editing of spot weld instructions	
14.2.8 Execution of instruction without current	
14.3 The SWI-function	14.11
14.3.1 General	
14.3.2 Function parameters	
14.3.3 I/O signals	
14.3.4 The spot weld instruction	
14.3.5 Programming the spot weld instruction	
14.3.6 Execution of instruction	
14.3.7 Repeat weld after interrupt	
14.3.8 Execution of instruction without weld current	
14.3.9 Reset of the weld controller	
14.3.10 Weld Controller programming.	



14.1 INTRODUCTION

For spot welding, a weld controller is necessary to control the weld sequence. The weld controller consists mainly of a timer unit and a power unit. This chapter describes how to program a weld instruction, both with a standard function, and with SWI (= Spot Weld Interface) function.

To simplify the programming of a spot weld instruction special, a control button **P (Process)** on the programming unit is used. An instruction which is a combination of a position instruction and a weld sequence instruction is obtained when this button is pressed.

14.2 THE STANDARD FUNCTION

14.2.1 General

The spot weld function controls an external welding controller with an instruction that contains a call for a sub-program. The physical communication between the robot and the weld controller is effected via a digital I/O board (DSQC 223). The signal sequence is defined in the sub program mentioned above, where ordinary logical instructions are used to set the outputs and to control the inputs that are necessary.

The programming of a spot weld instruction requires only one push on the programming unit P button. The number of the sub- program that is to be used is given in the instruction, and also any register that is intended for the step function.

14.2.2 Welding gun, manual operation

The GRIPPER function works, in the programmed instruction, by stating the number of the gripper and "grip" or "release" and the waiting time.

The spot weld function deals with two different cases of the GRIPPER function:

- 1 Only two welding guns.
The buttons for grippers 1 and 2 (grip, release) on the programming unit will open and close respectively the gun without any change in the display on the p-unit.
- 2 More than two guns.
GRIPPER 1 will open and close gun 1 without changes in the p-unit.
GRIPPER 2 will put the question GRIPPER NO = .

14.2.3 Function parameters

To get the welding function under the P-button, a special function parameter, SWELD, must be activated (SWELD = 1).

The function parameter SWELD is found in the MANUAL meny

See S3 Installation section 5.2.4.26.

14 Spot welding

The parameter ZONE is used to chose the zero zone size for the different types o fine points in the spot weld instruction.

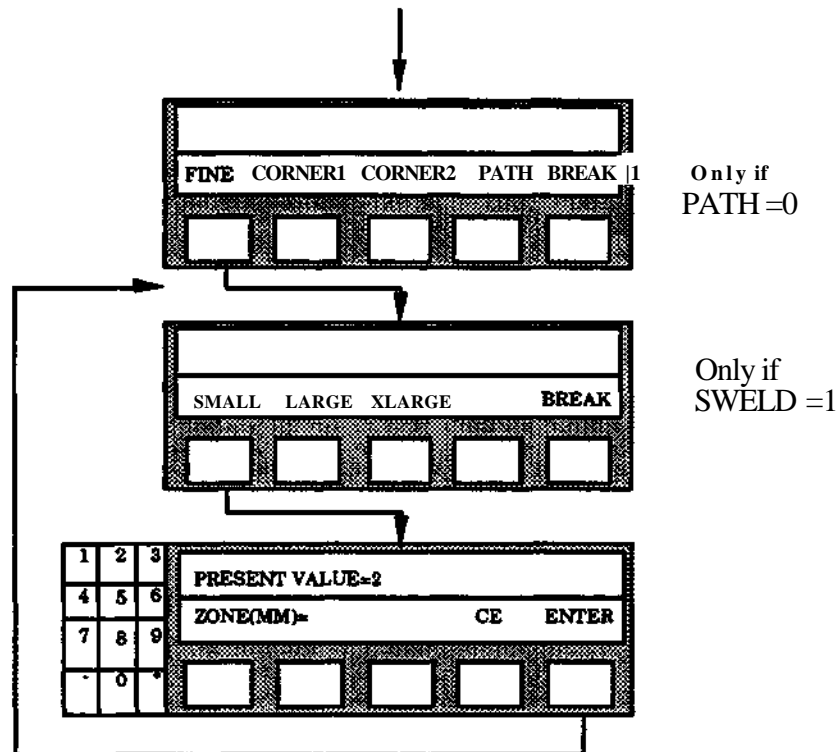
The valid sizes are:

Type	Def. range	default value
SMALL	2- 100 mm	2 mm
LARGE	2- 100 mm	10 mm
XLARGE	2-100 mm	20 mm

Note that the actual size of the zero zone will be vary throughout the working range. The size will correspond to the value stated in the parameters for movements in axis 1 with the arm fully stretched out. For other cases the size will be smaller.

(The figure describes how to set the parameter ZONE if SWELD = 1.)

MAN + SCAN + PARAM + CHANGE + SCAN + SCAN + SCAN + ZONE



14.2.4 The spot weld instruction

Here follows an example of what a standard instruction looks like:

```
100 POS V=100% FINE L WELDPR120 R5
```

Valid parameter values:

Zero zone size: FINE, FINE L, FINE XL

Program numbers: 100 - 9999

Register numbers: 0-119

Note.

In spot weld instructions, it is possible to program three different types of FINE arguments. This is necessary to allow the programmer to adjust the zero zone to the closing time of the gun.

Note.

Program numbers 0-99 are **not** allowed as subprogram numbers in the instruction. This is in order to avoid confusion with welding programs in the welding controller.

14.2.5 Programming of the first instruction.

The spot weld instruction is used for simultaneous programming of:

- * A fine point with optimal speed and accuracy.
- * A call of a subprogram which contains instructions for activation of the weld controller.

The spot weld instruction is programmed with the menu control button P. When the function is programmed it is possible to change the zero zone type and speed with the help of the corresponding function button **O- ZONE** and **V(%)**. For the function **O-ZONE** the argument is changed from large (FINE L) to extra large (FINE XL) and then to small (FINE) zero zone after each time the button is depressed.

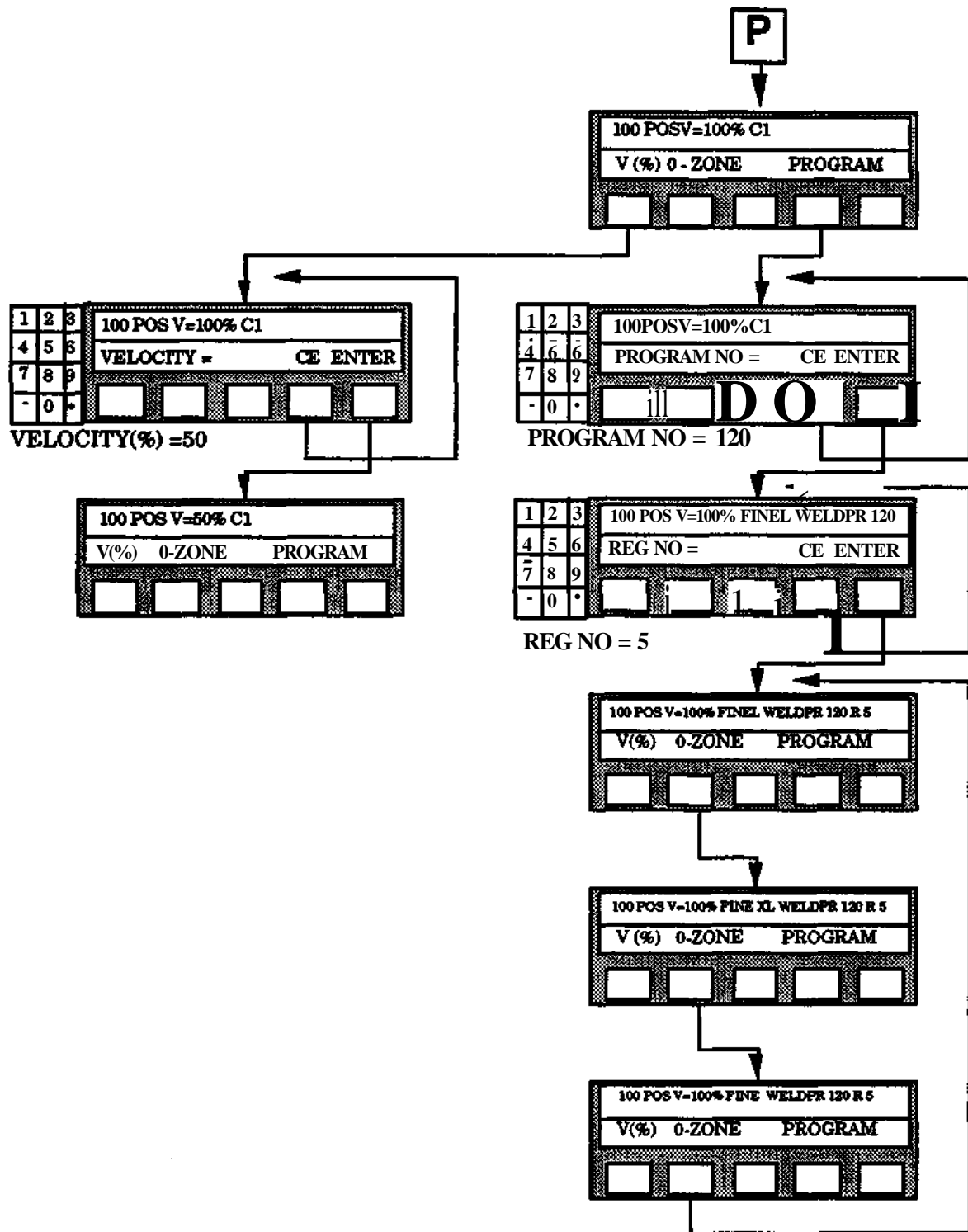
See the figure on the next page.

The positioning speed is expressed in % (0.1-799.9 %) of the programmed basic speed.

Procedure

- 1 Position the robot to the required weld point.
- 2 Press the P-button.
- 3 Press the **PROGRAM** button and select sub-program number (100-9999) and **ENTER**.
- 4 If a register is to be used for any step function press the button **REG NO** and select a number (0-119) and press **ENTER**. If the welding program is written as a pattern program with several returns, the register is used to ensure that the right sequence in the pattern program is executed.*)
- 5 The first spot weld instruction is now programmed.

*) The register is **not** a compensation register and the register is not incremented automatically (see pattern program in the section 11.4).



14.2.6 Subsequent spot weld instructions

The same argument which applied for the preceding spot weld instruction is repeated for the subsequent spot weld instruction in the program (i.e. each time the P-button is depressed). If required, it is always possible to change an argument through pressing the corresponding function button **V(%)**, **O-ZONE** or **PROGRAM**.

Procedure

- 1 Run the robot to the next weld point.
- 2 Press the P- button.

The instruction number and the same sub-program number and argument as in the preceding spot weld instruction are presented on the upper line of the programming unit display. If no further changes are necessary, the instruction is now completely programmed.

Otherwise change the arguments by pressing (V%), O-ZONE or PROGRAM.

14.2.7 Editing of spot weld instructions

A spot weld program often contains a large number of spot weld instructions with the same argument. To save editing time when the same argument is to be changed at the same time in a number of such instructions, the function button **MODARG** under the control button f- g has the following function:

When the button is used for a spot weld instruction, the same argument which applies for the preceding edited instruction is obtained. After the first spot weld instruction has been modified, only the function button **MODARG** needs to be used for the following spot weld instructions to obtain the same argument as in that first edited. Change spot weld instructions as follows:

Procedure

- 1 Press f- g.
- 2 Call the first spot weld instruction to be changed with the function button **INST NO** or **STEP**.
- 3 Press **SCAN**.
- 4 Press **MODIFY**.
- 5 Press **MODARG**.

Change the argument(s) to be modified **V(%)**, **O-ZONE** or **PROGRAM** by pressing the corresponding function button.

NOTE! When **MODARG** is depressed, the same argument is obtained as for the previous edited spot weld instruction.

14.2.8 Execution of instruction

14 Spot welding

During the execution of a spot weld instruction the following happens:

- 1 The robot is positioned to the programmed weld point.
- 2 When the robot TCP is within the zero zone selected, the selected subprogram is executed in the same way as an ordinary sub-program.
- 3 If a register number is programmed in the instruction, the selected part of the subprogram is executed in the same way as a pattern program. The register is not automatically incremented. This function gives the opportunity to successively change the welding parameters depending on the number of spots (step function).

When executing a spot weld instruction backwards, only the movement of the robot is performed.

14.3 THE SWI FUNCTION

14.3.1 General

SWI (Spot Welding Interface) is a software function in the robot system that controls the communication with the external weld controller. Programming is similar to that for the standard function. The instruction states which weld program is to be used and which register is to be used for control of the opening time of the gun. Subprograms for control of the weld controller are not necessary.

The concept also presents the possibility of using a double gun. In that case it is necessary to specify in the instruction which of the guns is to be activated (gun 1, gun 2, or both).

When a spot weld instruction is executed, the signal communication between weld controller and gun is performed automatically in a predetermined order.

The physical communication between the robot and weld controller is accomplished via a digital I/O board, (DSQC 223). Supervision and control of the gun is also performed through this board.

The function is developed to suit weld controllers that can use the same signal interface, like BOSCH PSS 208 IB, SATTWELD 1000 and others. It is also prepared for adaptation to weld controllers with a different signal interface. It is possible to read and edit a subset of the BOSCH weld controller parameters by use of serial communication.

14.3.2 Function parameters

To obtain access to the SWI function a special function parameter must be activated. The parameter SWELD, which gives access to the spot weld function under the button P on the programming unit, also has to be activated.

The function parameter SWELD is found in the MANUAL menu.
See Installation S3 section 5.2.4.26 - 27.

SWELD = 1 or 2 gives access to the parameters SWI, SERIAL COMM BOSCH and 64 PROGRAMS.

The digital I/O board DSQC 223 can be located in any slot and receives I/O numbers in the ordinary way. When the function parameter SWI is activated, a question appears regarding which board position (1- 6) is to be used.

The standard function with subprograms for control of the weld controller can also be used when the parameter SWI is activated.

Note.

If the SWELD parameter is reset, the parameter SWI is also reset.

14.3.3 I/O signals

See also Installation S3, section 3.13, for description on how to use I/O signals!

Output signals:

START 1	Start signal to the weld controller. The signal is automatically activated at execution of the SWI instruction. If a double gun is used, the signal is activated when welding is ordered for gun 1.
START2	Start signal to the weld controller if a double gun is used and welding with gun 2 is ordered.
CURRENT ENABLE	This signal is set low by the manual command INHIB, for testing of the robot program without weld current. The input signals FLOW OK, TEMP OK and CURRENT OK will not be supervised. The output signal WELD POWER will also be set low.
WELD POWER	High signal when the robot is in the MOTOR ON provided the signal CURRENT ENABLE is high. Low signal in all other cases. The signal can be used to control the weld power supply.
RESET	A pulse with a length of 50 ms is obtained when switching from MOTOR of to MOTOR ON and at PROG ST. It can be used to reset certain weld controllers after errors. Reset can also be activated manually.
GRIP1	Controls the work stroke of the gun. Equivalent to the corresponding system I/O signal.
GRIP 2	Controls the opening stroke of the gun. Equivalent to the corresponding system I/O signal
PARITY	Parity bit for program number (uneven parity). The signal is activated automatically when an SWI instruction is executed.
WELD PROGRAM	5 or 6 outputs to choose the weld program to be activated in the weld controller, depending on the definition of the parameter "64 programs". Programs 0-31 or 0-63 are permitted. For example, at execution of a SWI instruction with weld program 26, the outputs 10,12 and 13 ("2" + "8" + "16") will be set (Board in slot 1).

Input signals:

WELD READY	High signal from the weld controller when the weld is ready. The signal need to be low before START 1 or START 2 is set for the next spot weld. If not, error message 550 WELD ERROR 1 is displayed and program execution stops. If WELD READY is not recived within 20 sec. after START 1 or START 2, error message 550 WELD ERROR 2 is displayed and program execution stops (if SWI=1, MOTOR OFF is also received).
TIMER OK	High signals to indicate that weld controller is OK. As low signal, error message 551 WELD ERROR TIMER is displayed. SWI=1 Supervised during MOTOR ON. Low signal disables program execution and gives MOTOR OFF. SWI=2 Supervised when WELD READY is high. Low signal stops program execution. At PROG ST, the SWI instruction is repeated.
CURRENT OK	High signal to indicate that the weld current is within permissible tolerances. At low signal, error message 552 WELD ERROR CURRENT is displayed. SWI=1 Supervised during WELD POWER high. Low signal disables program execution, and gives MOTOR OFF. SWI=2 Supervised when WELD READY is high. Low signal stops program execution. REPEAT of weld is possible.
FLOW OK	High signal to indicate that the water supply is OK If the signal is low for more than 5 sec. error message 553 WELD ERROR FLOW is displayed. SWI=1 Supervised during WELD POWER high. Low signal for more then 5 sec.stops program execution and gives MOTOR OFF. SWI=2 Supervised when WELD READY is high. Low signal for more than 5 sec. stops program execution. At PROG ST next instruction is executed.
TEMP OK	High signal to indicate that the temperaure is OK At low signal, error message 554 WELD ERROR TEMP is displayed. SWI=1 Supervised during WELD POWER high. Low signal stops program execution and gives MOTOR OFF. SWI=2 Supervised when WELD READY is high Low signal for stops program execution. At PROG ST next instruction is executed.

ENABLE MOVE High signal indicates that the gun(s) is(are) open. Low signal is used to prevent robot movements when the gun is closed. The signal is checked after the programmed opening time has expired. If low after programmed opening time, register 98 is incremented by 1. If the signal is low more than 5 sec, error message 555 WELD ERROR ENABLE MOVE is displayed and robot movement is blocked (If SWI=1, MOTOR OFF is received).

Note! Reserved input signals that are not used must be strapped to 24 V. This does not apply to WELD READY which has to be supplied from the weld controller (see above) to make it possible to execute the SWI-instruction.

14.3.4 The spot weld instruction

Example of a spot weld instruction:

```
100 POS V = 100% FINE L WELDPR 20 R5 G12
```

Permitted parameter values:

Zero zone sizes: FINE, FINE L, FINE XL
predefined in the function parameters

Program numbers: 0-31 (0-63) give the SWI function
100 - 9999 give the standard function

Register numbers: 0-119

Gun arguments: (G1), G2 or G12

The instruction corresponds to the standard spot weld instruction with external weld controller. It is the programmed weld program number that determines whether the instruction should be executed as a standard or a SWI instruction.

The register in the SWI instruction contains the opening time for the gun in ms. If no register is programmed the opening time is set to 0. If the signal WELD READY is not given until the gun has been opened, it is not necessary to program any opening time in the instruction.

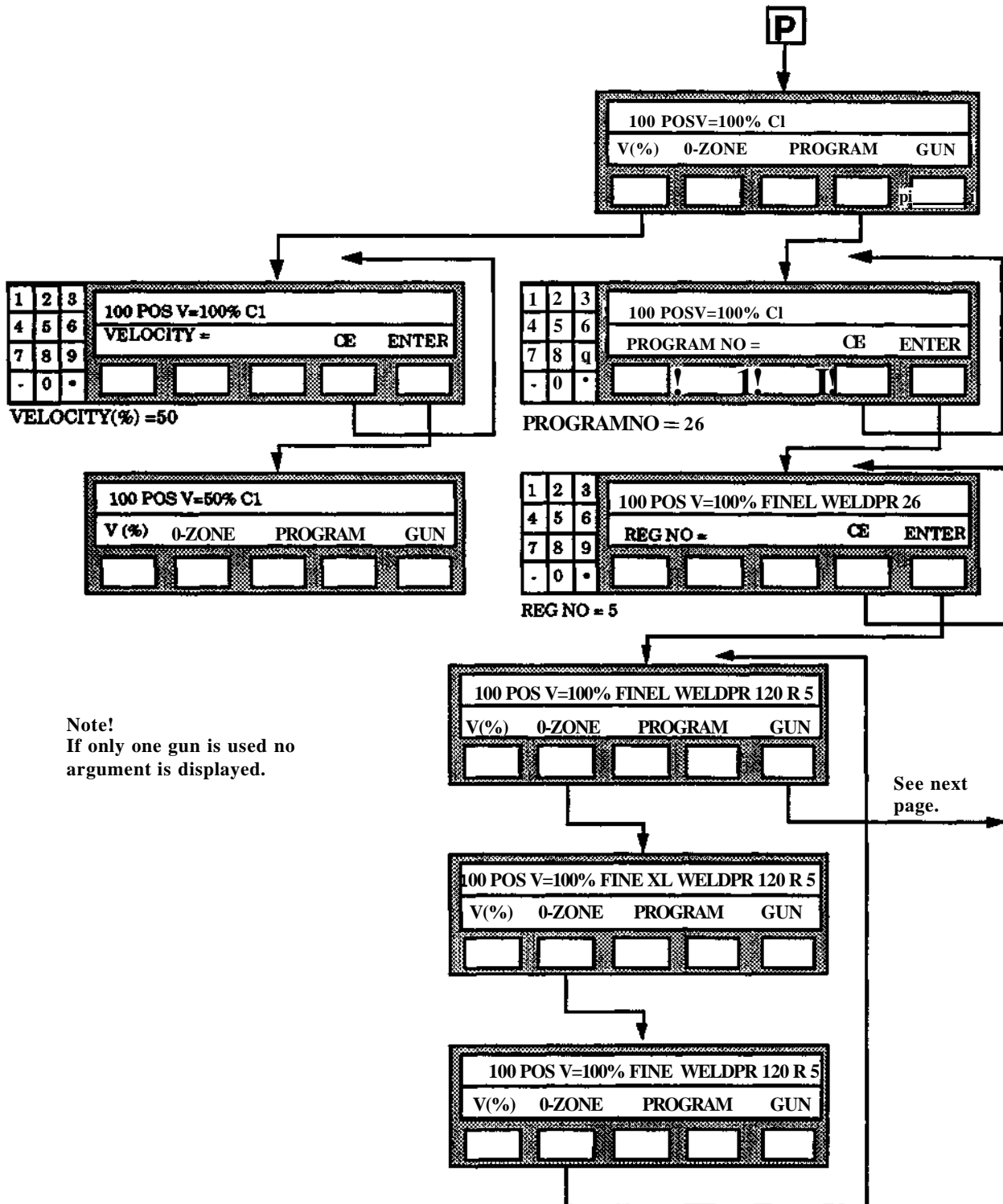
The reason why the opening time is set in a register and not directly in the instruction is that when trimming the opening time of the gun it is only necessary to edit one instruction.

When double gun is used, the last argument (GUN) states which of the guns is to be activated. If only gun 1 is to be activated, the argument is left out. If only gun 2 is to be activated, the argument G2 is displayed, and if both guns are to be activated, G12 is displayed.

Note! If the SWI parameter 1 or 2 is not activated, only standard function program numbers are possible. Gun argument is not possible to program.

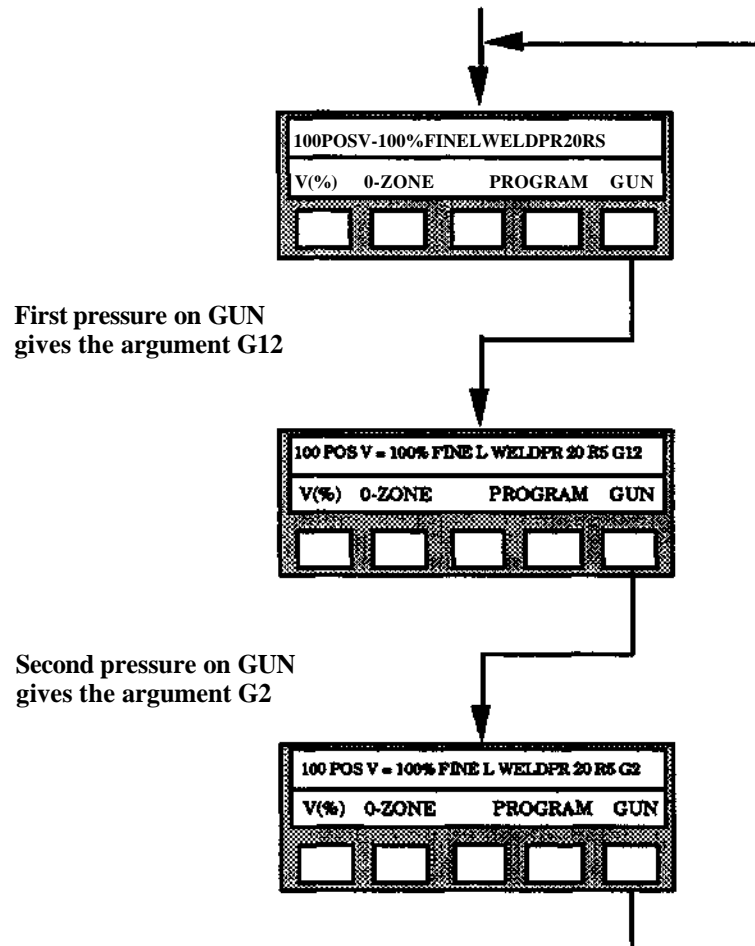
14.3.5 Programming the spot weld instruction

The spot weld instruction is programmed via the P button on the programming unit. When programming the very first spot weld all necessary arguments have to be specified. The subsequent spot weld instructions will get the same arguments automatically when pressing the P button.



The choice of gun when a double gun is used is effected as follows:

One press on the button gives the argument G12 (both guns activated), one more press on the button gives the argument G2 (gun 2 is activated). See fig. below.

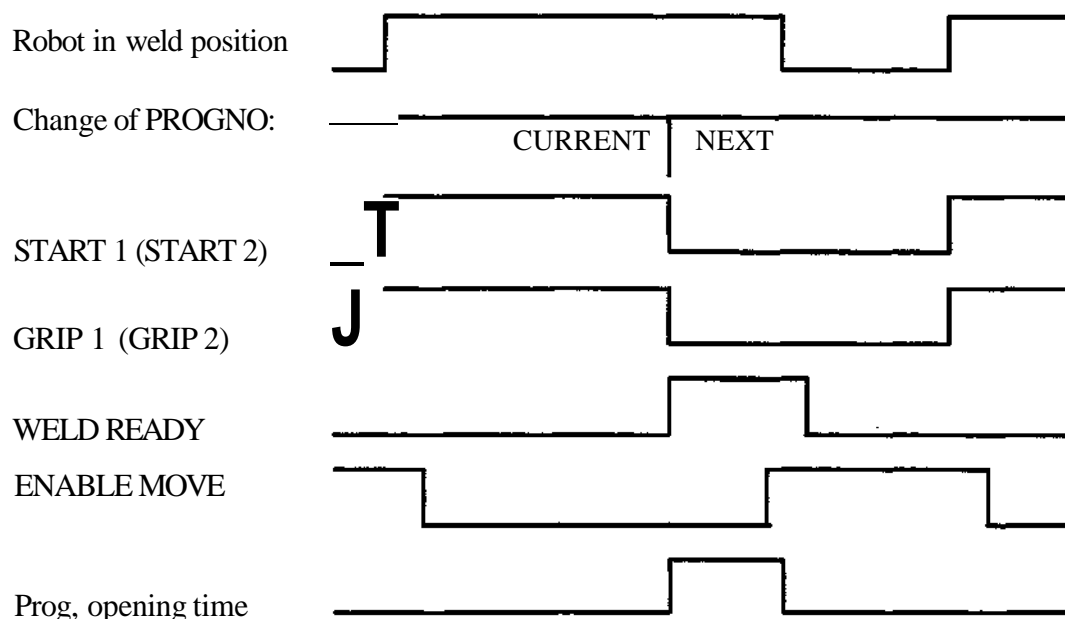


14.3.6 Execution of instruction

When executing the spot weld instruction, the robot will perform the following actions:

1. Outputs for program number and parity bit are set.
2. The robot moves to the programmed point.
3. Outputs for START 1 and GRIP 1 are set when the zerozone is reacted.
4. The robot waits for input WELD READY. If the waiting time exceeds 20 sec, START 1 and GRIP 1 are reset and program execution is stopped. If not, continues with 5.
5. START 1 and GRIP 1 are reset.
6. The internal timer for the opening time starts and spot counter (register 99) is incremented.
7. The instruction is now completed and execution of the next instruction is started. Any logical instruction will be executed directly. Another spot weld instruction will perform the following actions.
8. Outputs for program number and parity bit are set.
9. If input ENABLE MOVE is high, the robot moves to the next point. If ENABLE MOVE is low for more than 5 sec, the program execution stops.

A timetable of the signal sequences is shown below (principle):



Observe that there is a certain gain of cycle time if the WELD READY signal is received before the gun's opening phase is concluded. In such a case the opening time is used to perform the calculations for the next position. This is why the gun's opening time has to be specified in the instruction via the register number.

If the weld controller does not set the WELD READY signal before the gun has opened completely, the opening time is to be set to zero in the program. In such a case the register number can be omitted in the instruction.

Register 99 is used for counting the welded spots. Its content is incremented automatically after each spot. Resetting must be carried out through the user's program.

Register 98 is used for counting at how many spots the robot has been waiting for the signal ENABLE MOVE after the programmed opening time has expired. The register is therefore suitable for checking why the opening time of the gun has started to increase. Resetting of the register is performed in the user program.

When executing a spot weld instruction backwards, only the robot movements are performed.

14.3.7 Repeat weld after interrupt

If 550 WELD ERROR 1, 550 WELD ERROR 2 or 552 WELD ERROR CURRENT is received during weld process the question REPEAT? is displayed.

YES (or no answer) means that the interrupted instruction is executed again at PROG ST.

NO means that the next instruction is executed at PROG ST.

If 551 WELD ERROR TIMER the interrupted instruction is executed again at PROG ST.

If 553 WELD ERROR FLOW and 554 WELD ERROR TEMP, the program will start with the instruction that comes after the interrupted one at PROG ST.

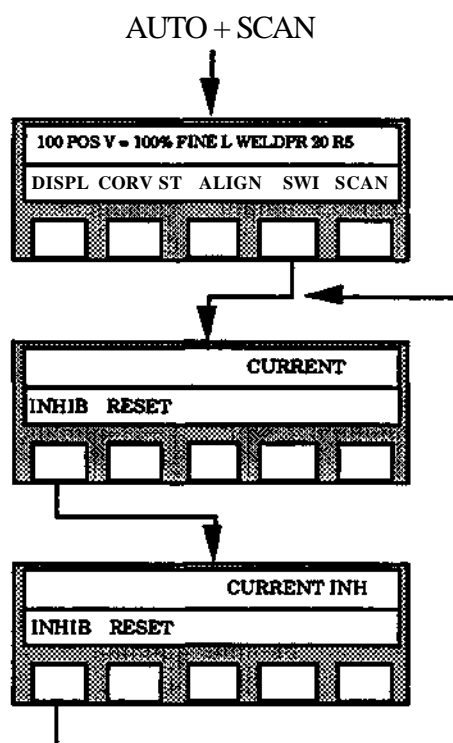
At Restart after power failure, the instruction is repeated if WELD READY was not received before the power failure.

14.3.8 Execution of instruction without weld current

In order to simplify the test procedure for a robot program with spot weld instructions, the program can be run without welding current. When this inhibit function is activated there is no supervision of FLOW OK, TEMP OK and CURRENT OK. The signals CURRENT ENABLE and WELD POWER will be set low.

When the INHIBIT function is deactivated CURRENT ENABLE is set high. WELD POWER will then be set high when the robot is in MOTOR ON mode.

The inhibit function is activated/ deactivated via a special SWI button under the AUTO menu. This button is only accessible if the SWI parameter is set (SWI = 1 or 2). The function is automatically deactivated if an INIT is performed. See figure below:



14.3.9 Reset of the weld controller

Manual reset of the weld controller can be done via the button RESET under the pushbutton SWI under the AUTO menu, (see figure above).

14.3.10 Weld Controller programming.

The serial interface to the weld controller Bosch PSS 2081 B is a RS232/V24 standard interface. The communication protocol is fixed to the controllers design. The parameters which can be programmed from the robot are a selection of the whole range of the timer parameters needed to control the weld process. The whole parameter set can be programmed from an external device supplied by the weld controller manufacturer.

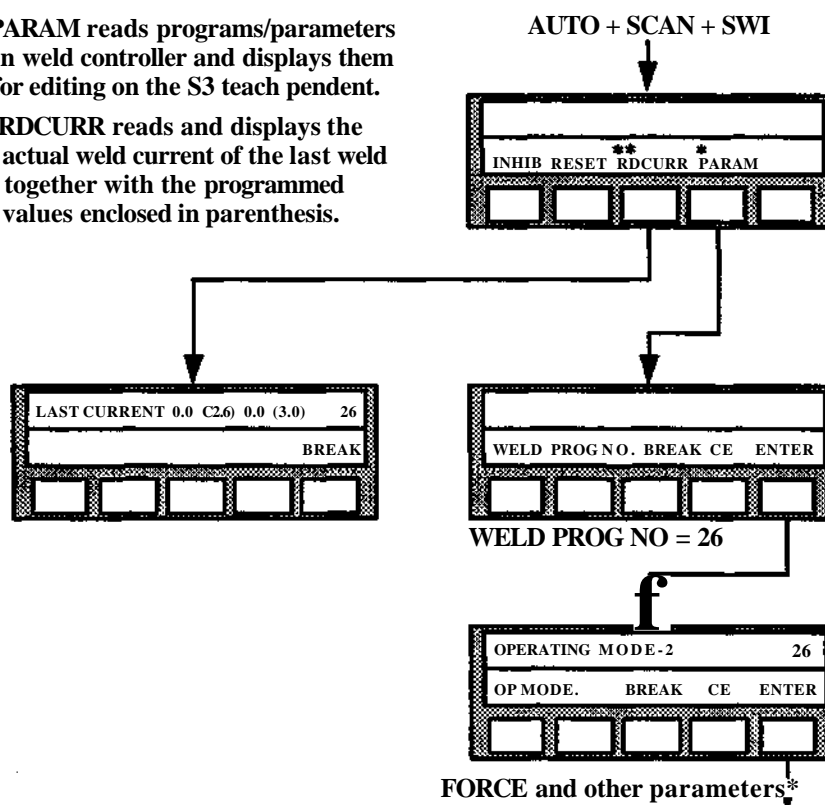
The parameters which can be changed from the robot are as follow.

Operating mode	0,1 or 2 PHA/SSU/KSR
Force	0,0 - 9,9 kN
Squeeze time	01-99 cycles
1st Weld time	01 - 30 cycles
1st Heat	00 - 99 scale values
1st Current	in kA
1st Pause time	00 - 99 cycles
Pulsation	01-09 pulses
Weld time	00 - 30 cycles
Heat	00 - 99 scale values
Current	in kA
Pause time	00 - 99 cycles
Hold time	01-99 cycles

With this interface it is possible to read the data block described above, to change the data and to send it back to the weld controller. Additionally, the actual current values of the last weld can be received by the robot control system and displayed together with the programmed values and in case of a weld controller error code can be received from the weld timer.

* PARAM reads programs/parameters in weld controller and displays them for editing on the S3 teach pendent.

** RDCURR reads and displays the actual weld current of the last weld together with the programmed values enclosed in parenthesis.



Parameters are also shown on the optional LCD of the S3 control system.

If **there** is no answer within 30 sec, an error message NO SERIAL CONTACT TIMER will be displayed.

ERROR LIST

3HAB 0008-2/Rev 1

1

General

When an error is detected, the ERROR lamp on the control panel illuminates. If a P-unit is connected to the robot system, an error message is presented at the same time in plain language on the upper line of the display. This applies also when a P- unit is connected after the error has developed and the ERROR lamp has illuminated.

Errors which can occur are divided into:

- Operational errors; the error messages beginning with a number 001-499.
- System errors; the error messages beginning with a number 501-999.

The appropriate corrective actions for the different errors are described below.

OPERATIONAL ERRORS

- 1 Check the error type on the upper line of the P-unit display.
- 2 If further information is necessary, go directly to point 6. Otherwise continue with point 3.
- 3 Press the control button SHIFT - the error message is then cleared.
- 4 Perform the control operation correctly so that the error status is not repeated.
- 5 Continue with point 8.
- 6 Seek the error message displayed in section 3.
- 7 Perform the appropriate actions stipulated in section 3.
- 8 If the same error message returns or persists, despite the incorrect operation not being repeated, contact service personnel.

SYSTEM FAULTS

- 1 Seek the error message presented in section 4.
- 2 Perform the appropriate corrective actions as described in section 4.
- 3 If the same error message persists or returns, call service personnel.

Note! Some error messages described in the sections 2-4 can occur for certain types of robots only.

2

Error buffer

Means: The system contains an internal error buffer, which can store 9 error messages. In the buffer the system stores:

- All kinds of error messages (both system errors and operator errors), occurred at the latest error occasion.
- Earlier messages about system errors.

The system can, on request, present the contents of the error buffer via:

- The programming unit. (Just error messages from the latest error occasion.)
- A printer, if the optional function Program printout is provided.
- A monitor, if this optional function is provided.

Facts: Display on the programming unit of error messages

The programming unit displays one error message at a time, according to the following:

- When the system stops running due to an error, the operator will see the first error message. An arrow on the display indicates if there are any consequent errors. In this case the operator can display these messages also, one by one.
- During manual operation the operator can, on request, display stored messages from the latest error occasion, one by one. The error messages are chronologically displayed.
- For most error messages, it's possible to get text in plain language by pressing "•" on the programming unit.

Display on the programming unit of error messages

The monitor displays all the messages within the error buffer, according to the following:

- When the system stops running due to an error, the operator will see all messages, occurred during that error occasion. The error messages are chronologically displayed.
- During manual operation the operator can, on request, display all stored messages. The error messages are in plain language and chronologically displayed.

Printout of error messages

- During manual operation the operator can, on request, display stored messages from the latest error occasion, one by one. The error messages are chronologically displayed.

Erasure of the contents in the error buffer and show text in plain language

- During manual operation the operator can, on request, erase all messages stored in the error buffer.
- On request, the operator can get text in plain language for most error messages.
- On request, the operator can load the texts in plain language from disk. (See Installation manual, S3.)

Example of a list of error messages

ERROR MESSAGE	CODE
***** : M c *****	
506 SERVO ERROR 2	1405->
506 SERVO ERROR 2	1605
536 ENABLE CHAIN FAULT	21
506 SERVO ERROR 2	1805
506 ENABLE CHAIN FAULT	21
504 PROGRAM RUN ERROR	7

Used: For fault tracing and for production follow-up.

Executed: The system reacts immediately after the procedure is concluded.

Procedures: Display on the programming unit of error messages (after production stop)
 Read the error message. Use the SHIFT button to display more error messages, if required.
 If you want to read messages about system errors once again, select ERRORS under the MANUAL menu.

3**Operational errors**

The following tables describe the errors which the operator can cause when programming or operating the robot system.

These are numbered from 001-499. The table gives.

- Error message
- Causes
- Recommended corrective action

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
1.NOT ALLOWED COMMAND	The button concerned is temporarily blocked by the system.HOLD can be active.	Select another button.
1.NOT ALLOWED COMMAND 10	Closed due to that input PROG STOP is active.	Deactivate input PROG STOP.
3J)ATA ERROR	Attempted entry of incorrect numerical value.	Depress ERASE and enter new numerical value.
4JNSTRNOT FOUND	After resequencing a program the system has detected jumps to non-existing instructions.	Check the program and correct the faulty jump addresses.
5.PROGRAM MISSING	Attempt to use a program number where instructions are missing with: <ul style="list-style-type: none"> • programming running • editing of complete program 	<ul style="list-style-type: none"> • Program the instruction under the program number concerned. • Select another program number.
6.MEMORY FILLED UP	Attempt to: <ul style="list-style-type: none"> • enter further instructions • copy program when the robot memory is full 	<ul style="list-style-type: none"> • 1 Clear the program block. 2 Erase superfluous instructions and programs. 3 Clear program block once again. • If the system includes a floppy disk unit, it is possible to: <ol style="list-style-type: none"> 1. Divide the existing program and store each section on a floppy disk under a separate program number. 2. Utilize the mass memory function for automatic exchange of the different program parts in the user memory.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
7.PROGRAMEND	Attempt to step past the last instruction	Call the first instruction number in the program concerned.
8.PROGRAM NUMBER OCCUPD2D	Attempt to copy a program under a program number where instructions are already stored.	Select another program number for the program copy.
9. INSTRUCTION NUMBER OVERFLOW	A.Attempt to program instruction with higher number than 65530. B. Attempt to resequence a program containing more than 6553 instructions.	A. Divide the existing program into a main program and subprograms. Copy the program and erase superfluous parts. Resequence and insert CALL-instructions. If available memory space is too small, use floppy disc. B. Avoid resequencing, or split the program according to point A above.
10. ADAPTIVITY ERROR1	A digital multi-bit sensors or analog sensor gives a signal outside its working area.	<ul style="list-style-type: none"> • Change one of the min. or max. -limits (or both) for the signal from the sensor concerned. • Correct current positions in the program.
10. ADAPTIVITY ERROR 2	A digital 2-bit sensor gives an illegal signal.	Check the sensor.
10. ADAPTIVITY ERROR3	A 1-bit sensor is used for direction searching or contour following.	<ul style="list-style-type: none"> • Select another sensor • Replace the sensor
10.ADAPTIVITY ERROR 4	A digital sensor with 2-8 bits is completely connected to the same group of digital inputs.	Change the connection according to the Installation manual.
10.ADAPTIVITY ERROR6	The correction movement has no direction or speed.	Determine direction and speed for the correction movement: <ul style="list-style-type: none"> • In a new instruction * By editing an existing instruction.
10.ADAPTIVITY ERROR 7	Sensor data for a sensor used missing.	Enter sensor data according to the Programming manual, chapter 9.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
IO.ADAPnVITY ERROR 8	The signal level programmed for search stop is outside the working range of at least one of the sensor.	Edit the instruction concerned so that the signal level for search stop is within the working range of the sensor.
11. OUTSIDE WORKING AREA 1-12	The robot axis 1 to 12 has been run outside its working area with the joystick.	Run the robot in the opposite direction with the joystick.
11. OUTSIDE WORKING AREA 20	Too large movement with axis 1,4 or 6 using the joystick.	Switch-off and start-up the robot system again.
11. OUTSIDE WORKING AREA 23	The angle between the lower and upper arms at axis 3 is too small or too large.	Leave this area using the joystick
11. OUTSD3E WORKING AREA 99	The main processor is in a temporary fault condition.	Re-initiate the system.
13 ROBOT NOT SYNCHRONIZED	A Some axis is NOT positioned if it is a robot with absolute measurement system. B Some axis is not synchronized.	A Re-start the system. Remedy possible fault. B Synchronize the system.
14AUTO MODE/ KEYLOCK	A The key-switch is in position AUTO and/or the key input on the remote control is activated. B Corresponding parameter is ACTIVE under AUTO C.	A Choose position MANUAL REDUCED SPEED or MANUAL FULL SPEED and/or deactivate the key input on the remote control. B Change to MANUAL REDUCED SPEED or MANUAL FULL SPEED mode or deactivate the parameter.
15. SENSOR NOT DEFINED	No sensor defined for sensor number entered.	Press ERASE and enter new sensor number.
16. SENSOR TYPE NOT ALLOWED	Incorrect sensor type for the adaptive function defined for the sensor number entered.	Press ERASE and enter new sensor number .

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION	
17. WRONG INSTRUCTION TYPE	Attempt to change: <ul style="list-style-type: none"> • Argument • Position • Speed for an instruction which does not contain positioning.	<ul style="list-style-type: none"> • Change the complete instruction. • Select another instruction number. 	
18. INSTRUCTION NUMBER OCCUPIED	Attempt to program instruction with occupied instruction number.	<ul style="list-style-type: none"> • Enter another instruction number. • Renumber the program and then enter an instruction number. 	
19. PROGRAM START	Attempt to step backwards and pass-by the first instruction in the program.	Erase the message and continue with the next action.	☾
20. SELECT ROBOT COORDINATES	Within this area the robot cannot be run in rectangular coordinates.	Select the robot coordinate system	☾
21. TCP NOT DEFINED	Attempt to program activation of an undefined TCP.	<ul style="list-style-type: none"> • Program another TCP. • Define the TCP that is to be activated via the instruction and try again. 	
23. ALIGN ERROR	Attempt to execute ALIGN with a too large angle deviation between the current tool orientation and the required one.	Turn the tool closer to the required orientation and try again.	
24. NOT DEFINED	(MH/ASM and GLUE) Attempt to execute ALIGN-FETCH on an undefined orientation register not containing a tool orientation.	<ul style="list-style-type: none"> • Select another orientation register. • Define the orientation by means of ALIGN-STORE. 	☾
25. INCOMPATIBLE OPTION	The required option can not be activated - because to that: A Another parameter under OPTION is already activated. B Required OPTION parameter can not be activated for the current robot type. SWI: System I/O or panel I/O are defined on current board.		☾

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
26 NOT ALLOWED COMMAND/ROOM FIXED TCP 1-8	<ol style="list-style-type: none"> 1 Can not change coordinate-system, when reference point is active. 2 Can not define a frame or pallet in wrist coordinate-system. 3 The function mirror is interrupted if the position is programmed with fixed TCP. Observe that reflection has been carried out on earlier positions with ordinary TCP. 4 MODPOS can not be carried out if a fixed TCP is active and the position instruction is made with an ordinary TCP and without external axes. 5 The instruction is blocked when fixed TCP is active. 6 It is not possible to define a base point with a room fixed TCP. 7 Not allowed to define a TCP automatic when a fixed TCP is active. 8 After programming an earlier pos-instruction a new TCP has been activated, so that the coordinate-system have been changed. The instructions are not of the same size, so it is not allowed to use SAME. 	<p>Deactivate reference point</p> <p>Activate an ordinary TCP(0-19)</p> <p>Use the function MODPOS on required positions.</p> <p>Erase the instruction and program it again, to convert to the wrist coordinate-system.</p> <p>Program in some other way.</p> <p>Activate an ordinary TCP.</p> <p>Program in the ordinary way.</p>
27 NO MORE INFORMATION	Required text in plain language not with the loaded texts	—
28 LOAD TEXT FROM DISK	Required text in plain language can not be displayed. The texts have not been loaded from the system disk.	Load the texts Menu: MANUAL + ERRORS
32 CALIBRATION ERROR 2	Fault during definition of EXTFRAME. Pos 1-3 does not describe a circle in space.	Choose 3 new positions and try again

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
32 CALIBRATION ERROR3	A Fault during definition of EXTFRAME. Calculated gear ratio for the ORBIT does not correspond with the ratio entered in the parameter memory. B The Radius is too small	A Check the gear ratio. B Make it bigger.
33 NOT ALLOWED EXTFRAME 1	When activating an EXTFRAME the system discovered that some "interfering" external axis is in an not allowed position for the specific EXTFRAME.	Align "interfering" external axes
33 NOT ALLOWED EXTFRAME 2	Attempt to activate an undefined EXTFRAME have been made or the robot system is trying to align with an undefined EXTFRAME.	Use only defined EXTFRAMEs
33 NOT ALLOWED EXTFRAME 4	Attempt to activate an EXTFRAME have been made while REFPOINT is active.	Deactivate the REFPOINT or don't activate the EXTFRAME
33 NOT ALLOWED EXTFRAME 5	Attempt to activate an EXTFRAME without activated ORBIT axis. The axis belongs to an inactive STATION or Attempt to define an EXTFRAME in the routine for calibration, that can't be activated because it belongs to an inactive STATION.	Activate the STATION which the axis belongs to.
33 NOT ALLOWED EXTFRAME 7	Attempt to deactivate a STATION while an EXTFRAME is active, where the EXTFRAME is an axis to the specific STATION.	Deactivate EXTFRAME first
33 NOT ALLOWED EXTFRAME 8	Attempt to align "interfacing" axes to a specific EXTFRAME have been made while these "interfacing" axes are not allowed to move (because they belong to inactive station or are blocked because the EXTFRAME in question is active).	Activate necessary STATIONS to activate blocked axes or deactivate EXTFRAME.

4

System faults

The errors which can occur in the robot system itself are described in the following table. These are numbered from 501 to 999.

The table gives.

- Error message
- Causes
- Recommended corrective measures.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE MEASURES
501 MEMORY FAULT RECORD	<p>A The system has discovered a summary check fault in the system parameter memory. Then the system has made a restart with the parameter basic values from PROM.</p> <p>B The system has discovered that the system parameters loaded has revision and version number separated from the ones in PROM. Then the system has restarted with the parameter basic values from PROM.</p>	<p>1 Insert the installation diskette, with the right system parameters, into the floppy disk unit.</p> <p>2 Choose the function button FR DISK</p> <p>3 The system starts-up.</p> <p>4 Press MAN/PARAM/RESOLV and check that the system has the correct RESOLV data.</p> <p>(Choose the function button PROM when restarting the system for the first time. Then define the installation system parameters including RESOLV data.) **)</p>
502 PROGRAM MEMORY FAULT	When restarting the system has discovered a check summary fault in the robot program memory. *)	Load the program block in question from DISK **)
503 EMERGENCY STOP	The robot has been stopped in an emergency.	<p>1. Correct the cause of the emergency stop.</p> <p>2. Press RESET on the control panel so that the emergency stop is cancelled. If the emergency stop cannot be cancelled:</p> <ul style="list-style-type: none"> • Check the safely board. • Start fault tracing, acc. to Chapter 4 in the Service Manual.

*) Causes to the discovery of check summary fault in system parameter memory or program memory:

- Too low battery voltage.
- Uncorrected or damaged battery.
- Damaged robot computer board.

**) Check the robot computer board incl. the memory battery back-up according to the Service Manual.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
503 WORKING RANGE STOP 2	One axis has been driven against a limit switch.	<ul style="list-style-type: none"> • Correct the cause of the stop. • Place the key-switch in position MAN. • Put the system back in operation by pressing the operation button and keeping that way and then depressing the dead man's handle. • Then drive the axis away from the limit switch with the joystick. <p>If the stop can not be restored:</p> <ul style="list-style-type: none"> • Check on not used safety circuits clamps. • Check the safety board. • Start fault tracing, acc.to Chapter 4 in the Service Manual.
504 PROGRAM RUN ERROR3	Attempt to start program execution when program is missing.	Enter a robot program from floppy disk or from programming unit.
504 PROGRAM RUN ERROR4	Attempt to execute a correction vector for another kind of instruction.	Call the correct instruction and try again.
504 PROGRAM RUN ERROR5	Last instruction was not RETURN nor JUMP in an executed program. Current program has been erased by the instruction ADD BLOCK	Insert the correct instruction.
504 PROGRAM RUN ERROR7	<p>A Attempt to execute a movement path outside the robot's working area.</p> <p>B The wrist centre of the robot or the TCP is in 20 mm range of the base Z-axis.</p>	<p>A Edit the movement path. Check the TCP location.</p> <p>B Use robot coordinates or modify the TCP track.</p>
504 PROGRAM RUN ERROR8	Attempt to call a sub-program or robot program without instructions.	<ul style="list-style-type: none"> • Edit the program number in the instruction. • Create a subprogram or robot program with this number

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 9	(MH/ASM AND GLUE) A Attempt to execute a weave movement with a weave subprogram where the first instruction is not a positioning instruction. B (AW) Attempt to execute a weave movement with either: <ul style="list-style-type: none"> • No positioning instruction after weave start. • The distance between weave start and next positioning instruction is zero. (AW). 	Edit weave program. Edit the program.
504 PROGRAM RUN ERROR 10	A Attempt has been done to execute an impermissible instruction. B (AW) Undefined welddata number. • Exchange welddata nr. C (not AW) Attempt has been made to execute a weave movement when instruction in sub weaving program is faulty.	A Check the program. B "Define present welddata. C Correct the sub weaving program.
504 PROGRAM RUN ERROR 12	Attempt to execute a jump to a non-existent instruction.	<ul style="list-style-type: none"> • Edit the jump address. • Enter an instruction with this number.
504 PROGRAM RUN ERROR 13	Attempt to execute a pattern subprogram, when the register concerned contains an incorrect value.	Check and change the value in the register concerned.
504 PROGRAM RUN ERROR 14	Attempt to execute a nested call to a 11th subprogram level (or a 12th level of interrupt program).	Edit the robot program.
504 PROGRAM RUN ERROR 15	The main processor is in a temporary fault condition.	Switch-off and start-up the robot once again. Check electrical disturbance on inputs. Check that input 6 and output 7 is not used as general I/O when glue is activated.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 16	<p>A Main program (program 0) is missing.</p> <p>B The active program replaced by the corresponding one on floppy disk at program loading with the subfunctional APPALL under the GET B-function</p> <p>C Another mass memory error</p>	<p>A, B Enter a main program.</p> <p>C Try to perform the corresponding GET B-function manually.</p>
504 PROGRAM RUN ERROR 17	<p>A. A positioning instruction after a circle point contains an illegal argument.</p> <p>B. Attempt to back to the start point after program stop in the middle of a circular arc.</p>	<p>A. Edit the instruction and repeat the attempt.</p> <p>B. Run if possible to the end point of the circle and back from this point instead.</p>
504 PROGRAM RUN ERROR 18	<p>The positions of the circle points are selected incorrectly so that either:</p> <ul style="list-style-type: none"> • The radius becomes too large. • Two or several points in the circular arc coincide. 	<ul style="list-style-type: none"> • Remove the circle point and run in straight lines instead. * Change the position of the points concerned so that a circular arc is obtained.
504 PROGRAM RUN ERROR 19	<p>The circular interpolation has been interrupted so that it cannot continue.</p>	<ol style="list-style-type: none"> 1. Erase the error message with the SHIFT key. 2. Check that there is no obstacle in the direct path of the robot forward to the next point. 3. Run instruction by instruction to the next point.
504 PROGRAM RUN ERROR 20 (only for arc welding robot)	<p>The acknowledgment of transferred data is omitted when executing the arc welding instruction EXTPOS.</p>	<p>Check the peripheral equipment concerned.</p>

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 24	A The robot is trying to reach a position with another configuration than it had when that position was programmed. This can only occur in rectangular coordinates.	<ul style="list-style-type: none"> • Modify the orientation in the current or preceding position or <ul style="list-style-type: none"> • Chose Modified Rectangular Coordinates or <ul style="list-style-type: none"> • Deactivate the configuration supervision (see 8.5, HANDCHK).
	B The axis 4 position differs more than 45 degrees or the axis 6 position differ more than 90 degrees from the positions they had in the next position instruction at programming time.	<ul style="list-style-type: none"> • Modify the orientation in the current or preceding position or <ul style="list-style-type: none"> • Chose Modified Rectangular Coordinates <ul style="list-style-type: none"> • Deactivate the configuration supervision (see 8.5 HANDCK)
504 PROGRAM RUN ERROR 26	A The robot is trying to reach a position with another configuration than it had when that position was programmed. This can only occur in rectangular coordinates together with FINE-positions.	<ul style="list-style-type: none"> • Modify the orientation in the current or preceding position or <ul style="list-style-type: none"> • Chose Modified Rectangular Coordinates or <ul style="list-style-type: none"> • Deactivate the configuration supervision (see 8.5, HANDCHK)
	B On the way towards the position, axis 4 deviates more than 45 degrees, or axis 6 deviates more than 90 degrees from the start position, in direction away from the stored configuration of the next position. This can only occur in rectangular coordinates.	<ul style="list-style-type: none"> • Modify the orientation in the current or preceding position or <ul style="list-style-type: none"> • Chose Modified Rectangular Coordinates or <ul style="list-style-type: none"> • Deactivate the configuration supervision (see 8.5, HANDCHK)
	C The time, programmed for the time positioning, is to short.	C Increase the programmed time.
	D Program running in rectangular coordinates is not possible.	D Change to program running in robot or modrect coordinates, or move the motion a little away from the singular point.
	E Weaving: the crosstime is to short compared to the amplitude.	E Increase crosstime or decrease amplitude.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 27	The value in a number register is not valid.	Check the program and edit the register handling.
504 PROGRAM RUN ERROR 50	Automatic definition of a program displacement is in progress. During the process, at least two positions approach each other too closely: <ul style="list-style-type: none"> • When the positions are selected. * In the resulting program displacement. 	Select three new positions and try again.
504 PROGRAM RUN ERROR 51	Attempt to activate an undefined TCP	<ul style="list-style-type: none"> • Activate another TCP. • Define required TCP and try again.
504 PROGRAM RUN ERROR 53	(MH/ASM and GLUE) Attempt to use: A An undefined gripper. B An output which is reserved for a gripper.	A Change the number of grippers in the function parameters. B Use another gripper/output.
504 PROGRAM RUN ERROR 54	(MH/ASM and GLUE) Attempt to execute a PALLET-instruction with an undefined pallet.	A Define the pallet. B Use an already defined pallet.
504 PROGRAM RUN ERROR 55	(MH/ASM and GLUE) Attempt to execute a PALLET-instruction with the register values pointing out a position outside the pallet.	Change the register values.
504 PROGRAM RUN ERROR 57	(MH/ASM and GLUE) Attempt to execute a TOOLREL-instruction with displacements/rotations outside permitted limits.	Change the register values.
504 PROGRAM RUN ERROR 58	Attempt to execute a gluing instruction with the parameter GLUE not activated.	Activate the GLUE function parameter GLUE under OPTION - provided that the robot system is a gluing system.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 62	(AW) Weaving data not defined for the current weaving data field.	* Select another weaving data field. • Define weaving data.
504 PROGRAM RUN ERROR 63	(AW) CROSSTIM is 0.	Define CROSSTIM.
504 PROGRAM RUN ERROR 64	(AW) BASEPoint not defined.	Define BASEPoint and try again.
504 PROGRAM RUN ERROR 65	(AW) Sensor data undefined for current sensor data field	• Select another sensor data field. • Define sensor data.
504 PROGRAM RUN ERROR 66	(AW) Internal error in calculation of end position for AUTO-SEARCH.	Edit the start position for AUTO-SEARCH and try again. If the error persists after several trials, check the computer board.
504 PROGRAM RUN ERROR 67	(AW) The robot velocity is too high at connection or disconnection of SPS.	Edit the program. Reduce the zero zone or the velocity.
504 PROGRAM RUN ERROR 68	No position stored in the current position register.	* Select another position register. • Store a position.
504 PROGRAM RUN ERROR 69	Internal system fault.	Re-initiate the system.
504 PROGRAM RUN ERROR 70	The soft-servo number or softness vector is undefined.	Active another set of softness, or define the relevant softness set under the manual menu.
504 PROGRAM RUN ERROR 71	The wrist centre point or the TCP has entered within 20 mm of the base Z-axis. This is not allowed in rect. or modrect. coordinates.	Use robot coordinates or modify the robot pass.
504 PROGRAM RUN ERROR 72	Incorrect TCP may be active. Indical to TCP x IS ACTIVE.	Check that the correct TCP is active at the programming unit. Continue program execution by pressing instruction start or program start once more.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 73	The weaving amplitude cannot be reached because the TCP is too close to the x-axis (weaving axis) of the wrist.	<ul style="list-style-type: none"> • Change TCP • Reduce the amplitude.
504 PROGRAM RUN ERROR 74	The robot moved when the activation/deactivation of a station was requested.	Stop the robot and repeat the attempt.
504 PROGRAM RUN ERROR 75	Timeout in supervision of a spot weld. WELD READY not received within 5 s after START 1/2.	Check welding controller.
504 PROGRAM RUN ERROR 76	Output used in SWI function not defined.	Define the function parameter for GRIPPER3.
504 PROGRAM RUN ERROR 77	Execution of an SWI function without activated SWI parameter.	Define the SWI parameter or change the instruction.
504 PROGRAM RUN ERROR 78	The program number in the instruction is not permitted.	Change the program number in the instruction.
504 PROGRAM RUN ERROR 80	Attempt to reach a position with a wrong TCP active have been made. Pos-instructions and stored positions can not be used between fixed TCP and ordinary TCP.	Check the program and active TCP.
504 PROGRAM RUN ERROR 81	Attempt to switch coordinate-system with a TCP-instruction have been made, when the reference point is active.	The reference point has to be deactivated before change of coordinate-system is possible.
504 PROGRAM RUN ERROR 82	Attempt to execute an instruction have been made, who don't function together with a fixed TCP.	The instruction in question can only be executed with a ordinary TCP active.
504 PROGRAM RUN ERROR 84	Attempt to activate an undefined LOAD.	Activate another LOAD. Define required LOAD and try again.
504 PROGRAM RUN ERROR 85	Incorrect LOAD may be active. Identical to "LOAD x IS ACTIVE"	Check on teach pendant, that the correct LOAD is active. Continue program execution.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
504 PROGRAM RUN ERROR 86	Robot coordinates are not allowed during arc welding.	Change coordinate system
504 PROGRAM RUN ERROR 87	Max. allowed velocity for an external axis limits the velocity of the robot during the arc weld process.	Reprogram so that no external axes limits the velocity of the robot.
504 PROGRAM RUN ERROR 88	The instruction can not be restarted.	Make ordinary restart or move the robot.
504 WRONGLY PLACED I/O BOARD	No digital I/O-board defined in place stated by the parameter BOARD POSITION.	Check the function parameters for the I/O and SWI.
505 SERVO ERROR 1	The servo computer does not accept an order from the main computer, because of a serious system fault.	Check the computer board.
505 SERVO ERROR 8001	The axes computer has not accepted an order from the main computer, due to a serious error in the main computer.	Re-initiate the system. If the fault persists.replace the main processor board.
506 SERVO ERROR 2 1101-1124	Resolver fault, channel x. Fine resolver, axis 1(1101)-12(1112) Coarse resolver, axes 7(1119)-12(1124)	A Check for any fault by measuring resistance in wiring and resolver. (resolver~20ohm) B Replace the serial measurement board or the axis board.
506 SERVO ERROR 2 1201-1224	Resolver error, channel y. Fine resolver, axis 1(1201)-12(1212) Coarse resolver, axis 7(1219)-12(1224)	See error type 1101 - 1124

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
506 SERVO ERROR 2 1301 • 1312	<p>Speed error. The motor for axis 1 (1301)-12 (1312) runs considerably faster than is commanded from the control system. Cause, internal axes:</p> <p>A Considerable interference between the axes at high speeds results in incorrect speed at certain points in the user's program.</p> <p>B Incorrectly commutated motor</p> <p>C Axis stops (the motor receives no current)</p> <p>D Incorrect acceleration of the motor.</p> <p>Cause, external axes:</p> <p>E See point A above.</p> <p>F Incorrect acceleration of the motor.</p>	<p>A Lower the programmed speed locally in the program where the error occurs and/or change the positioning pattern so that the axis which trips because of the incorrect speed is not so active in this part of the program.</p> <p>B Check that the correct commutator offset is entered for the axis. If the axis has been repaired, remeasured and enter a new offset</p> <p>C 1. Check all fuses. 2. Check the motor current using the test outlets. Check the wiring, the drive unit and the robot computer board.</p> <p>D Replace the robot computer or drive unit.</p> <p>F 1. Check external controller, drive stage and external wiring. 2. Replace robot computer or external axis board.</p>
506 SERVO ERROR 2 1401-1412	<p>Jam error, internal axes</p> <p>The motor for axis 1 (1401) - 7(1407) remains stationary despite the robot computer commanding current to the motor. Cause.</p> <p>A The robot has run against an obstacle.</p> <p>B. Overload.</p> <p>C. The motor receives current but is incorrectly commutated.</p> <p>D. The motor does not receive current.</p> <p>E. Resolver fault.</p> <p>F. Motor fault.</p> <p>G. Mechanical fault.</p>	<p>A 1. Select operation mode RUN and run away from the obstacle. 2. Remove the obstacle or edit the program.</p> <p>B. Check that the load and its lever arm do not exceed the specified maximum limits.</p> <p>C. Check that the correct commutator offset is entered for the axis. If the axis has been repaired measure and enter the correct offset.</p> <p>D 1. Check all fuses. 2. Check the wiring, the drive unit and the robot computer board.</p> <p>E. Turn the axis with the system in the MOTOR OFF mode and check if resolver fault is indicated.</p> <p>F. Measure motor data.</p> <p>G. Check, with the system in the MOTOR OFF mode that axis movement is free in its complete working range.</p>

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
	Jam error, external axes.	A 1. Select operation mode MOTOR ON and run the axes away from the obstacle. 2. Remove the obstacle or edit the program.
	The motor for external axis 7 (1407) -12 (1412) runs slower than 2 % despite the robot computer commanding a higher speed. Cause: A The axis has run against an obstacle. B The axis receives no torque. C Axis not correctly adjusted.	B 1. Check that any brake is released. 2. Check that a speed reference is received from the axis board. If unrepairable, replace the axis board. 3. Check the external controller, driver, wiring and motor. C. Increase the KP-value for the axis. The KP-values are defined in the system parameters.
506 SERVO ERROR 2 1500	Resolver fault, supply	1. Check for any fault by measuring resistance in wiring and resolver. (resolver-20ohm) 2. Replace the serial measurement board or the axis board.
506 SERVO ERROR 2 1601-1612	Too high speed in TEST-position for axis 1 (1601) - axis 12 (1612). The speed protection is tripped when the programming unit is connected and out and the key switch not in position 100 %. Cause: A Chain fault related to an emergency stop. B External interference on the robot or the external axes. C Low motor torque. D Incorrectly adjusted external axes 7-12.	A Correct the primary fault. B Reduce the effect of external forces C See "506, SERVO ERROR 2 1401-1407", item C and D. D Check that the adjustment of the external axes does not cause any heavy speed overshoots.
506 SERVO ERROR 2 1701-1707 (Concerns only robots with tachometer).	Resolver/tacho signal differs for axis 1 (1701) - axis 7 (1707). Cause: Tachometer signal indicates another speed than that calculated from the resolver signal.	1. If new tachometer has been installed, check that positive voltage is obtained if the axis is turned in a positive direction. 2. Check that the voltage reaches the axis board. 3. Replace the axis board.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
506 SERVO ERROR 2 1801-1812	<p>Position error with brake activated for axis 1(1801)-12(1812). One of axes 1-12 moved though the system has ordered the brakes to be applied. Cause:</p> <p>A Manual release of the brakes and movement of the axis. B Abrupt braking, with the brake, from high speed, by the Dead Man's Handle or a switch-over to MOTER OFFC</p>	<p>A,B Check that all axes are in safe positions for program start or synchronizing. If not, run the robot to a safe position using the programming unit. C Change the brake. Poor brake.</p>
506 SERVO ERROR 2 2001-2024	<p>Resolver error channel x or y axes 1(2001)-12(2012) Coarse resolver, axes 7(2013)-12(2024)</p>	<p>A Check for fault by measuring resistance in wiring and resolver (resolver-20 ohm) B Replace the serial measurement board or the axis board.</p>
506 SERVO ERROR 2 2101-2112	<p>Racing protection at start-up. The racing protection for axis 1(2101)- axis 7(2107) has tripped as the axis has not remained stationary when the RUN status was activated the first time after initialization. Cause:</p> <p>A Incorrect commutation offset B Faulting tachometer (B concerns only robots with tachometer). C Faulty resolver D Low motor torque</p>	<p>A 1. Check that the correct commutator offset is entered. 2. If the commutation has been changed because of repairs, make new measurements. See Installation manual. B 1. If the tachometer is newly installed, check that positive voltage is obtained when the axis is turned in a positive direction. 2. Check that the voltage is received at the axis board. C If the resolver is newly installed, check turning the axis very slowly in a positive direction and reading with the help of the programming unit that the resolver value increases (when reaching 8191, the value restarts from 0). D 1. Check the drive unit. 2. Check the resistances of the motor windings.</p>

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
	Servo lag error for axes 1 (2101) - 12 (2112) when running in the TEST mode. Cause: Major positioning error because the axis has not rotated in accordance with the received position references.	(See "error code 1401-1421") (See "error code 1301-1312")
506 SERVO ERROR 2201-2224	Resolver error, channels x and y. Pine resolver axes 1(2201) -12(2212) Coarse resolver, axes 7(2213) - 12(2224)	A Check for fault by measuring the resistance of the wiring and the resolver (resolver~20 ohm). B Replace the serial or the serial measurement board or the axis board.
506 SERVO ERROR 223XX	Error in the revolution counter of axis XX. Axes 1-6 are supervised by measurement board 1 and axis 7 by board 2. All revolution counters on the measurement boards are uncalibrated.	Check the measurement system, especially all resolver and measurement board connections. Recalibrate the robot revolution counter. <i>Check the robot calibration position. If the problem persists, replace the serial measurement board</i>
506 SERVO ERROR 2240X	To high speed on axis X.	Reduce the speed.
506 SERVO ERROR 25104	Incorrect input data. Cause: Parameters entered are outside specified limits.	Check that all parameters entered are within the limits specified.
506 SERVO ERROR 25105-5111 5120-5123 5130-5136 5137-5138 (The numbers shows where in the software the stop occurred.)	A External electrical equipment has jammed the robot computer or serial measurement board. B Faulty computer board or serial measurement board. C Faulty wiring between the robot computer board and the serial measurement board.	Re-initiate the system and start-up again. If the error persists, check the wiring or replace the computer board and/or the serial measurement board.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
506 SERVO ERROR 2 5150	Voltage drop from rectifier	A Check fuses 8 Check that voltage to rectifier
506 SERVO ERROR 2 7001	The maximum permissible waiting time for answer to the main computer from the axis computer has expired, depending on: <ul style="list-style-type: none"> • An earlier error has caused the servo computer to halt. • Fault in the robot computer. 	Find and correct the earlier error then re-initialize the system. If the error persists, change the robot computer board.
506 SERVO ERROR 2 7002	Non-valid channel index. Communication fault between the diagnostic software and the main computer, caused by a fault in the control program.	Internal error. Re-initiate the system.
506 SERVO ERROR 2 7004	Erroneous sync position. The sync position of the robot is not correct, because there are no values for the sync position in the robot control program.	Internal error. Re-initiate the system.
506 SERVO ERROR 2 7005	Fault in internal calculation routine in the control program.	Exchange the control program.
506 SERVO ERROR 2 7006	Execution of a program module in the main computer starts, before a valid acknowledge signal has occurred.	Check the computer board. If no error is found, change the control program.
506 SERVO ERROR 2 7007	STALL ALARM. The main computer has detected that messages from the axis computer are not transmitted fast enough.	Check the computer board. If no error is found, perform a common error tracing, according to Chapter 4 in the Service Manual.
506 SERVO ERROR 2 7008	The servo computer has detected an undefined error in: <ul style="list-style-type: none"> • It's own program. • The main computer program. 	Check the computer board.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
506 SERVO ERROR 2 7009	Maximal waiting time for answer to the main computer from the servo computer, for respond on a movement segment, has been exceeded.	Re-initialize the system,
506 SERVO ERROR 2 7010	The servo computer has given an unallowable error code to the main computer.	Re-initialize the system.
506 SERVO ERROR 2 8001	The servo computer has not approved an order from the main computer due to a serious error in the main computer.	Re-initiate the system. If the fault remains, replace the control program.
506 SERVO ERROR 2 9000	Transmission fault between the serial measurement board and the robot computer board. Causes: A External disturbances B Wiring fault C Electronic fault	A Check the installation B Check and measure the wiring C Change the serial measurement board or the robot computer board.
507 JOYSTICK ERROR	1 Indicates a fault in the joystick function. 2 Erroneous coordinate system button.	If the fault persists, check the joystick according to the Service Manual.
508 DISK MEMORY FAULT 1	Error in the floppy disk memory.	<ul style="list-style-type: none"> • Repeat attempt • Change disk
508 DISK MEMORY FAULT 2	Data stored has disappeared.	<ul style="list-style-type: none"> • Repeat attempt • Change disk
508 DISK MEMORY FAULT 3	Data stored incorrect	<ul style="list-style-type: none"> • Repeat attempt • Change disk
508 DISK MEMORY FAULT 4	A. Floppy disk faulty B. Floppy disk not formatted.	A. Repeat attempt B. Change disk
508 DISK MEMORY FAULT 6	Floppy disk with storage protection.	<ul style="list-style-type: none"> • Remove storage protection • Change disk
508 DISK MEMORY FAULT 7	Floppy disk unit not ready.	Insert disk correctly <ul style="list-style-type: none"> • Close hatch • Change disk

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
508 DISK MEMORY FAULT8	Erroneous data at attempt of storing data on floppy disk.	<ul style="list-style-type: none"> • Try again • Change disk
508 DISK MEMORY FAULT10	Internal Winchester fault	Re-initiate the system
508 ERROR IN TEXT PROGRAM 18	Not possible to load the block into the reserved memory area. The block contains other instructions than comment instructions or is corrupt.	Load the block into the user memory and check the contents. Edit the block or select a block from the system disk and try again.
508 TEXT PROGRAM TOO BIG 18	Not possible to load the block into the reserved memory area. More than 25% of the block is used.	Load the block into the user memory and check the contents. Edit the block or select a block from the system disk and try again.
508 DISK MEMORY FAULT20	Attempted loading of function parameters from wrong disk.	<ul style="list-style-type: none"> • Change disk
508 DISK MEMORY FAULT 21	Attempt to program storage when the whole floppy disk is occupied.	<ul style="list-style-type: none"> • Change disk
508 DISK MEMORY FAULT22	Attempt to load program from an empty floppy disk.	<ul style="list-style-type: none"> • Replace disk
508 DISK MEMORY FAULT24	Attempt to load a program block not available on the floppy disk.	<ul style="list-style-type: none"> • Select another program block number. • Change disk.
508 DISK MEMORY FAULT 25	Attempt to add a program block that is larger than the user memory.	<ol style="list-style-type: none"> 1. Erase all programs in the user memory (program numbers 0- 9999). 2. Load required program block, with the function ADDALL under the MANUAL menu. <p>If this does not work, split the program block into two parts at the robot, where the block was originally programmed.</p>

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
508 DISK MEMORY FAULT26	An attempt has been made to add a program block as the space available in the user memory is insufficient.	<ol style="list-style-type: none"> 1. Erase any superfluous programs in the user memory. 2. Store the remaining part of the program block as block X. 3. Load the program block containing the required program. 4. Erase all unwanted programs. 5. Add block X with the function ADDALL. <p>If this is not successful, use the instruction GET B in the program for alternate running of program blocks X and N.</p>
508 DISK MEMORY FAULT 27	Error in the stored program block.	<ul style="list-style-type: none"> • Select another block number. • Change disk.
508 DISK MEMORY FAULT 28	Fault when formatting floppy disk.	<ul style="list-style-type: none"> • Change disk • Try again
508 DISK MEMORY FAULT29	Attempt to load a non-existing subprogram from floppy disk.	<ul style="list-style-type: none"> • Select another block number. * Select another program number.
508 DISK MEMORY FAULT30	Floppy disk function is faulty.	Re-initiate the system.
508 DISK MEMORY FAULT31 (for arc welding robots only)	Weld data missing for the program block concerned.	<ul style="list-style-type: none"> • Select another block number • Change disc.
508 DISK MEMORY FAULT 32 (for arc welding robots only)	Incorrect version of weld data.	Select disc with right weld data version.
509 SYNC ERROR	Error when the robot system is to be synchronized.	If repeated attempts are not successful, perform a common error tracing, according to Chapter 4 in the Service Manual.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
509 SYNCHRONIZATION ERROR 10XX	<p>Robot axis XX is outside its outside the working range</p> <p>Error has occurred in the control system, axis XX.</p>	<p>Move the axis manuell or with the joystick after pressing RESYNK (AUTO+SCAN+SCAN) into the work, range. Switch power off and the on.</p> <p>If the problem persists, it might be caused by DSQC 234, see the Service Manual.</p> <p><i>Check the robot calibration position before returning the robot to production!</i></p>
509 SYNCHRONIZATION ERROR 11XX	Working range defined for external axis no. XX is too great with the gear ratio in use.	Redefine the working range under PARAM or check that the correct resolver configuration has been selected.
509 SYNCHRONIZATION ERROR 14XX	Resolver on axis no. XX not calibrated.	
509 SYNCHRONIZATION ERROR 15XX	Resolver error. Axis no. XX has obtained an impermissible resolver value with absolute measurement.	Check the measurement system.
509 SYNCHRONIZATION ERROR 16	System fault in absolute measurement system.	See instructions in Service Manual for checking the robot system.
509 SYNCHRONIZATION ERROR 17	<p>The robot is too far away to restart.</p> <p>The robot program running can not be restarted directly.</p>	<p>Start-up the system the conventional way. <i>Check the robot calibration position before returning the robot to production!</i></p>
509 SYNCHRONIZATION ERROR 20XX (concerns only robots in the 8000-serie).	Robot axis xx is outside the working area.	Manoeuvre the axis into the workingarea and switch off the system. Try once more.
509 SYNCHRONIZATION ERROR 21XX	The revolution counter for axis XX not calibrated.	Check measurements system, battery. Battery voltage should exceed 7.0V.
509 SYNCHRONIZATION ERROR 22XX	Error on the revolution counter on axis XX. The revolution counter has lost its value.	Check the measurement system. The robot remains unsynchronized. Operation mode is blocked until the error is remedied.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
509 SYNCHRONIZATION ERROR 23XX	Error on the revolution counter on axis XX. The expected revolution counter value does NOT correspond to that of the serial measurement board revolution counter.	Check the measurement system, especially all resolver and measurement board connections. Recalibrate the robot revolution counter. <i>Check the robot calibration position. If the problem persists, replace the serial measurement board.</i>
510 SYSTEM FAULT3	Internal fault in the control program module in programmed robot running.	Switch-off and start-up the robot system once again. If the fault persists exchange the control program.
510 SYSTEM FAULT4	A. The internal restart was interrupted undefined. B. Power failure in an instruction that can't be restarted.	A,B Switch off the system and start in the conventional way.
510 SYSTEM FAULT5	Restart data in memory is wrong.	Re-initiate the system.
510 SYSTEM FAULTS	An error has occurred in the error buffer. All old messages have been erased.	
510 SYSTEM FAULT7	An unwanted jump in the program may have occurred.	Internal error. Check active instruction before resume program execution.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION																								
513 WRONGLY PLACED I/O BOARD X (X=subcode which defines the board position concerned)	A Board missing on position X. B The board on position X does not correspond to the board-type specified in the system parameters (specified manually for subcode 1-6). C The board on position X erroneous.	A Place a board on position X (or change the system parameter for subcode 1-6). B Place the correct type of board (or change the system parameter for subcode 1-6). C Check the board.																								
	<p>The board positions in question are coded for the board type, that is specified in the system parameters (manual specifications for the 6 I/O positions). The board positions are coded in accordance with the following:</p> <table> <tr> <th>Subcode</th><th>Type</th><th>ace to param</th></tr> <tr> <td>0</td><td>Safety board</td><td></td></tr> <tr> <td>1</td><td>I/O board, position 1</td><td></td></tr> <tr> <td>2</td><td>I/O board, position 2</td><td></td></tr> <tr> <td>3</td><td>I/O board, position 3</td><td></td></tr> <tr> <td>4</td><td>I/O board, position 4</td><td></td></tr> <tr> <td>5</td><td>I/O board, position 5</td><td></td></tr> <tr> <td>6</td><td>I/O board, position 6</td><td></td></tr> </table>	Subcode	Type	ace to param	0	Safety board		1	I/O board, position 1		2	I/O board, position 2		3	I/O board, position 3		4	I/O board, position 4		5	I/O board, position 5		6	I/O board, position 6		
Subcode	Type	ace to param																								
0	Safety board																									
1	I/O board, position 1																									
2	I/O board, position 2																									
3	I/O board, position 3																									
4	I/O board, position 4																									
5	I/O board, position 5																									
6	I/O board, position 6																									
514 COMMUNICATION ERROR 1 (apply only to robots equipped with computer link)	Insufficient space in the robot buffer memory for data from superior computer.	Restart the robot system.																								
514 COMMUNICATION ERROR 2 (apply only to robots equipped with computer link)	Data fault in message received from superior computer.	Check superior computer.																								
514 COMMUNICATION ERROR 3 (apply only to robots equipped with computer link)	Transmission error on the data link.	Repeat the attempt.																								
514 COMMUNICATION ERROR 4 (apply only to robots equipped with computer link)	Insufficient space in the robot memory when attempting to load a program from a superior computer	Erase any superfluous program in the robot memory.																								

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
514 COMMUNICATION ERRORS (apply only to robots equipped with computer link)	Program with program number selected not in the robot memory.	Select a program with another program number.
514 COMMUNICATION ERROR 6 (apply only to robots equipped with computer link)	Maximum waiting time before acknowledgment from superior computer exceeded.	Check superior computer.
514 COMMUNICATION ERROR 8 (For robots with computer link only)	Computer link not implemented.	Check that function parameter C LINK is set (See Inst. manual).
514 COMMUNICATION ERROR 9 (For robots with computer link only)	Fault in vision system.	Restart the vision system. If the fault persists, check the connection or contact service personnel.
514 COMMUNICATION ERROR 10 (For robots with computer link only)	Vision timeout.	Restart the vision system. If the error persists, check the connection or contact service personnel.
514 COMMUNICATION ERROR 11 (For robots with computer link only)	Incorrect robot mode.	Turn MOTORS OFF.
514 COMMUNICATION ERROR 12 (For robots with computer link only)	Insufficient space in robot memory when an attempt is made to load configuration data.	Check that the block to be loaded is intended for the robot in question.
514 COMMUNICATION ERROR 13 (For robots with computer link only)	Incorrect check sum in configuration data loaded. Robot system initialized with configuration data from PROM.	Make another attempt to load the configuration data.
514 COMMUNICATION ERROR 14 (For robots with computer link only)	Incorrect weld data version.	Configure to right weld data version on superior computer.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
515 TIMEOUT ERROR (only arc welding robots)	When programming the arc-welding instruction EXTPOS, the response from the external equipment is not received before the maximum permissible waiting time has been exceeded.	Check the external equipment,
516 OVERRIDE ERROR 1 (only arc welding robots)	Override has been defined as: A. Welding is not performed. B. The welding data number has been changed.	A. Wait until welding is performed. B. Try again.
516 OVERRIDE ERROR 2 (only arc welding robots)	The limit values for override definition has been exceeded.	Store the values and keep defining override values.
517 WELD ERROR 1010 (only arc welding robots)	The current supervision has detected that current has disappeared for longer than 500 ms.	Check the thread feeder and current supply.
517 WELD ERROR 1020 (only arc welding robots)	The gas/coolant supervision has detected that gas/coolant is missing.	Check the gas flow and coolant flow.
517 WELD ERROR 2010 (only arc welding robots)	The welding arc has not been lighted before the indicated time from welding start order.	Check thread feeder and current supply. Change the maximum time in the start data.
517 WELD ERROR 2020 (only arc welding robots)	Gas and coolant flow received has not been started before 2 seconds after the robot has moved to the position for welding start	Check gas flow and coolant flow,
518 RIO COMMUNICATION ERROR 1	No communication with host processor (PLC).	Check hardware.
518 RIO COMMUNICATION ERROR 2	Bad communication with host processor.	Check hardware.
518 RIO COMMUNICATION ERROR 3	Na-chip on RIO board is out of function.	Check hardware.
523 ERROR STATION DRIVE UNIT 1 (only arc welding robots)	At power-on the activation of manipulator 1 failed (max. allowed waiting time for acknowledgment signal from manipulator 3 seconds exceeded).	Try again. If repeated attempts are not successful, see Service Manual.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
523 ERROR STATION DRIVE UNIT 2 (only arc welding robots)	The robot moved when activation/deactivation of a station was requested.	Stop the robot and repeat the attempt.
523 ERROR STATION DRIVE UNIT 3 (only arc welding robots)	An input indicates that the drive units of a station have been activated, in spite of the fact that no such order has been given on the output.	Check that the station connection is in accordance with the function parameter STATION. In that case check the external equipment,
523 ERROR STATION DRIVE UNIT 4 (only arc welding robots)	An input indicates that the drive units of a station have been deactivated, in spite of the fact that no such order has been given on the output.	Check the cables and that the station connection is in accordance with the function parameter STATION. In that case check the external equipment.
523 ERROR STATION DRIVE UNITS (only arc welding robots)	An output for activating a STATION is set, but no answer from corresponding input has been given within 5 sec.	Check the the connections from ORBIT to the robot control system.
523 ERROR STATION DRIVE UNIT 6 (only arc welding robots)	Signals for activation to STATIONS have been reset at an earlier stop. These signals have automatically been set high again.	Check that the connected stations are relevant. If that is the case, restart the program again, otherwise connect the STATIONS in question.
524 OUTSIDE WORKING AREA 7-12	External axis is outside working area.	Modify the program.
536 ENABLING DEVICE FAULT 1 536 DMH FAULT 1 (only IRB 1500)	A The enabling device has been held in an intermediate position B A fault in the contacts of the enabling device.	Press the enabling device again. If the fault remains, check the contacts under the enabling device.
536 SPEED SUPERVISION FAULT 5	The key switches position and reported status of the servo computer's SPEED signal do not coincide.	Examine the SPEED signal status on the back panel between the safety board and the robot computer board.
536 RUN RELAY FAULT 13	Control signal and check signal to/from the operation contactor do not coincide.	Check the operation contactor and the cabling.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
536 ENABLE CHAIN FAULT 21	<p>The ENABLE chain has been broken as a result of one of the following fault:</p> <p>A Drive unit/fatal fault</p> <p>B Rectifier/fatal fault</p> <p>C Measurement system/fatal or nonfatal fault</p>	<p>Check the LED on the front of the safety board.</p> <p>A The LED is lit:(nonfatal fault) The occurred fault is automatically reset. The enable chain is closed after a few seconds. The system can be in operation mode again.</p> <p>B The LED is put out:(fatal fault) The occurred fault has to be manually reset. The enable can only be reset by an system initiation. Search for the cause to the fault and remedy it. Do a system initiation.</p>
536 SAFETY HAZARD 33	Fault in the safety board indicated via the supervision for the ENABLE loop.	Replace the safety board.
536 SAFETY HAZARD 34	An attempt has been made to start-up the system with undefined commutating offset.	Define the commutating offset.
536 RUN BUTTON FAULT 35	The run button has been pressed for 30 seconds.	Check and if needed remedy the run button or the cabling.
536 KEY STATUS FAULT 36	The position/status of the key is incorrect updated e.i. because of two key turns in a row, when the system was in operation or the run chain was broken.	Change the position of the key or re-initiate the system
536 MOTOR OVER-LOAD FAULT 101-107	Overload in motor 1-7. Certain models: No axes number is stated.	<p>Re-start the system.</p> <ul style="list-style-type: none"> • Check wiring • Reduce duty cycle
538 AUTO MODE STOP 538 WORK STOP 1	One or several work stop switchers connected to the system have been activated.	<ul style="list-style-type: none"> • Check for the cause to the stop and remedy any possible problem in the system. • Then reset the activated switchers.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
538 GENERAL MODE STOP 2 538 SAFETY STOP 2	One or several safety stop switchers connected to the system have been activated.	<ul style="list-style-type: none"> • Check for the cause to the stop and remedy any possible problem in the system. • Then reset the activated switchers. If the stop can not be reset: <ul style="list-style-type: none"> • Check the clamps that are not used by the safety circuits. • Check the safety board. • Perform a common error tracing according to the Service manual.
539 INTERNAL RECTIFIER POWER FAULT 1	No power from the internal rectifier.	Check if possible clamps on not used internal rectifier. Check and if needed remedy rectifier and cabling.
539 EXTERNAL RECTIFIER POWER FAULT 2	The signal POWER OK is missing.	Check connection for POWER OK. Check and if needed remedy rectifier and cabling.
539 WARNING/ RECTIFIER PHASE MISSING 3	At least one phase in the power supply to the rectifier.	Check and if needed remedy fuses and cabling.
540 BRAKES RELAY FAULT	System error which means that the brakes are not released normally.	Examine the connections to the robot brakes and locate the break in the connection.
542 MOTORTYPE NOT DEFINED	Attempt has been made to take the system in operation without that the motortype has been defined.	Define motortype according to the Installation manual S3.
544 WARNING NEW RESOLVER DATA LOADED	The system has discovered a difference between the new resolver values, and the resolver values already in the parameter memory.	Choose if the system is to be started on the new or old resolver data. (Remaining system parameters according to earlier executed operation). Check that the robot has the right system parameters after re-start.
547 HYDRAULIC PRESSURE ERROR 1 (ROBOTS IN THE 8000 SERIES)	Faulty pressure in the Z axis pressure unit.	Check the pressure unit.
548 ERROR IN RUN CHAIN 2	Run chain 1 is correct but run chain 2 indicates a fault	Check the run chains.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
550 WELD ERROR 1	WELD READY not reset when START 1 or START 2 is activated.	Check the welding controller.
550 WELD ERROR 2	WELD READY not received within 20 s after START 1 or START 2 has been activated.	Check the welding controller.
551 WELD ERROR TIMER	Signal TIMER OK is missing.	Check the welding controller.
552 WELD ERROR CURRENT	Signal CURRENT OK is missing.	Check the welding controller.
553 WELD ERROR FLOW	Signal FLOW OK is missing for more than 5 sec.	Check the cooling water supply.
554 WELD ERROR TEMP	Signal TEMP OK is missing.	Check the welding controller.
555 WELD ERROR ENABLE MOVE	The system has been waiting for signal ENABLE MOVE more than 5 sec. upon gun opening.	Check the welding gun and sensors.
560 ERROR/ROBOT COMPUTER BOARD xxxx	A robot computer board component is malfunctioning. See chapter 5 for further information.	See subcode in chapter 5.
562 ERROR/ROBOT DRIVE UNIT xxxx	Erroneous signals from drive unit after start-up. See chapter 5 for further information.	See subcode in chapter 5.
563 ERROR/DRIVE UNTTOFFSx	Drive unit exceeds limit values when compensating for offset. See chapter 5 for further information.	Check the drive unit connection. If all connections are correct, replace the drive unit. If this doesn't work, change the robot computer board.
	Subcode x: Unit:	
	1 Drive unit 1	
	2 Drive unit 2	
	3 Drive unit 3	
	4 Drive unit 4	
	5 Drive unit 5	
	6 Drive unit 6	
	7 Drive unit 7	
567 ERROR/SYSTEM BOARD xxxx	Data from the safety board is incorrect. See chapter 5 for further information.	See subcode in chapter 5.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
570 ERROR/MEASURE SYSTEM, AXES 1-6:xxxx	The measurement system diagnostics has discovered an error. See chapter 5 for further information.	See subcode in chapter 5.
571 ERROR/MEASURE SYSTEM, AXES 7:xxxx	The measurement system diagnostics has discovered an error. See chapter 5 for further information.	See subcode in chapter 5.
572 ERROR/MEASURE SYSTEM, AXES 8-12 xxxx	The measurement system diagnostics has discovered an error. See chapter 5 for further information.	See subcode in chapter 5.
573 ERROR/MEASURE SYSTEM, AXES 7-12 xxxx	The measurement system diagnostics has discovered an error. See chapter 5 for further information.	See subcode in chapter 5.
580 WARNING DISK xxxx	Error when starting disk. See chapter 5 for further information.	See subcode in chapter 5.
581 WARNING COMPUTER LJNE zzzz	Error when starting computer link. See chapter 5 for further information.	See subcode in chapter 5.
582 WARNING PRINTER xxxx	Error when starting printer. See chapter 5 for further information.	See subcode in chapter 5.
583 WARNING SENSOR xxzx	Error when testing the main computer serial links. See chapter 5 for further information.	See subcode in chapter 5.
585 WARNING PROG UNTTxxxx	Error when starting the programming unit. See chapter 5 for further information.	See subcode in chapter 5.
586 WARNING MONITOR xxxx	Error when starting monitor. See chapter 5 for further information.	See subcode in chapter 5.
588 WARNING WINCHESTER xxxx	Error when starting the Winchester memory. See chapter 5 for further information.	See subcode in chapter 5.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
589 RECTIFIER MISSING	The configuration check of the rectifier indicates type code 0, that means no rectifier connected.	Check that: <ul style="list-style-type: none"> • rectifier is connected correctly
590 RECTIFIER WRONG TYPE	Wrong type of rectifier.	Check the rectifier with respect to robot type and change to the right type.
591 DRIVE SYSTEM ERROR	The drive system has indicated error, but no error status is placed in either rectifier or drive unit.	Re-initiate the system.
592 RECTIFIER OUTPUT VOLTAGE HIGH	The voltage from the rectifier is too high.	Change rectifier,
593 RECTIFIER TEMP. HIGH	The rectifier unit is overloaded.	Check if the surrounding temperature is too high. Reduce the intermittence during the run mode.
594 SHUNT RESISTANCE TEMP. HIGH	The return supply power from the robot motors is too high. Too high surroundings temperature and too high input voltage to the rectifier can also cause too high temperature.	Check the input voltage to the rectifier. Check the surroundings temperature around the control cabinet. Reduce the intermittence during run mode.
595 DRIVE UNIT MISSING 1-7	The configuration check of the drive unit for the axis in question, indicates type 0, that means no drive unit connected.	Check if the drive unit on the axis in question is in the right position. Check the flat cable between drive unit and computer.
596 DRIVE UNIT WRONG TYPE X	The drive units on the axis X in question are of the wrong type.	Check the type of drive units on the axis in question, and change to the right type.
597 DRIVE UNIT CURRENT HIGH X	The drive unit for the axis X in question has gone into the over current limit.	Check that the cabling out of the drive units or the motor winding is short-circuit. Check that the out-put on the drive unit isn't shortcircuit.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
598 DRIVE UNIT CURRENT ERROR X	The current out from the drive unit for the axis X in question do not correspond to the ordered value.	Check that the cables out from the drive unit in question are unbroken. Probable error cause is interruption in one of the cables or the motor winding.
599 DRIVE UNIT TEMP. HIGHX	<p>The temperature on the out-put level on the drive-unit in question for the axis X in question has exceeded.</p> <p>The axis number can in some systems not be displayed.</p>	<p>Check:</p> <ul style="list-style-type: none"> * if the robotaxis for the axis in question do not work slowly * the commutation offset * that the surroundings temp, around the control cabinet is below the max. allowed <p>Reduce the intermittence during run mode.</p>
602 VISION SYSTEM ERROR	The hardware for VISION is not available.	Otherwise see separate documentation for VISION.
700 WARNING NEW RESOLVER DATA LOADED	The system has detected a difference between down loaded resolver data and data that earlier was in the parameter memory.	<p>Choose if the system shall start with new or old resolver data. (Other systemparam. ace. to previous operation).</p> <p>After start check that the robot has correct system parameters.</p>

5

Diagnostic messages

Diagnostic error concerns the fault- and warning message the system discovers when testing the hardware during the start-up sequence.

Depending on the type of malfunction, two types of message may be displayed: error message or warning message.

Error message:

When this type of message is displayed on the programming unit, the system is always interrupted and put in an error mode. If the malfunction occurs during start-up, this is interrupted.

Warning message:

The printout is given after a malfunction has been detected in any of the units, but the malfunction is not serious enough to interrupt the system.

Normally, all message printouts are displayed on the programming unit display, but during the diagnostic test at start-up, no text can be displayed during certain periods of time. The error codes may then be displayed by the error LED on the computer board front. See chapter 6 for a detailed description.

Error and warning messages using a subcode:

560 ERROR/ROBOT COMPUTER BOARD xxxx
 562 ERROR/DRIVE UNIT xxxx
 567 ERROR/SAFETY BOARD xxxx
 570 ERROR/MEASUREMENT SYSTEM, AXES 1-6 xxxx
 571 ERROR/MEASUREMENT SYSTEM, AXIS 7 xxxx
 572 ERROR/MEASUREMENT SYSTEM, AXES 8-12 xxxx
 573 ERROR/MEASUREMENT SYSTEM, AXES 7-12 xxxx
 580 WARNING DISK xxxx
 581 WARNING COMPUTER LINK xxxx
 582 WARNING PRINTER xxxx
 583 WARNING SENSOR xxxx
 585 WARNING PROGRAMMING UNIT xxxx
 586 WARNING MONITOR xxxx
 588 WARNING WINCHESTER xxxx

for which xxxx represents one of the following subcodes, more thoroughly describing the cause of the message.

The following is the list of the subcodes which may be displayed along with an error or warning message.

SUBCODE	CAUSE	RECOMMENDED CORRECTIVE ACTION
0001	Error at main computer instruction test	Replace the robot computer board
0002	PROM checksum error	a Replace the EPROM b Replace the robot computer board
0003	RWM error	Replace the robot computer board
0004	Code downloading error	Replace the robot computer board
0008	Main computer error when handling not allowed command	Replace the robot computer board
0009	Error in test 1 of communication between I/O and main computers	Replace the robot computer board
0010	Error in test 2 of communication between I/O and main computers	Replace the robot computer board
0011	Error at ending of I/O computer first test sections	Replace the robot computer board
0013	Error when testing the main computer sensors, serial channels	Replace the robot computer board
0014	Error when testing the enable chain	a Replace the robot computer board b Replace voltage supply unit DSQC 241 c Replace the computer rack rear plane
0017	Error when starting the I/O computer later test sections	Replace the robot computer board
0018	Error in test 1 of communication between servo and main computers	Replace the robot computer board
0019	Error in test 2 of communication between servo and main computers	Replace the robot computer board
0039	Servo computer communication error	Replace the robot computer board
0040	Error when starting the servo computer later test sections	Replace the robot computer board
0070	Error when testing the main computer serial channel. Out = low, In = high	Is used at manufacturing check of robot computer board
0071	Error when testing the main computer serial channel. Out = low, In = high	Is used at manufacturing check of robot computer board

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
0072	Error when testing the main computer serial channel A	Is used at manufacturing check of robot computer board
0075	Error when testing the I/O computer serial channel A, computer link	Is used at manufacturing check of robot computer board
0076	Error when testing the I/O computer serial channel B, printer	Is used at manufacturing check of robot computer board
0080	Error when testing the I/O bus	Is used at manufacturing check of robot computer board
0081	Erroneous data from the I/O bus	Is used at manufacturing check of robot computer board
0101	Error at I/O computer instruction test	Replace the robot computer board
0102	PROM checksum error, I/O computer	a Replace the EPROM b Replace the robot computer board
0105	Erroneous I/O computer "stall timer"	Replace the robot computer board
0106	Erroneous I/O computer" time out timer"	Replace the robot computer board
0107	Erroneous I/O computer" interval timer"	Replace the robot computer board
0108	I/O computer error when handling not allowed command	Replace the robot computer board
0109	Error in communication between I/O and main computers	Replace the robot computer board
0112	Internal diagnostics error in I/O computer	Replace the robot computer board
0114	Error when testing the enable chain	a Replace the robot computer board b Replace voltage supply unit DSQC 241 c Replace the computer rack rear plane
0116	Error when initiating the I/O computer	Replace the robot computer board
0120	Error when testing printer output	Replace the robot computer board if printer is to be used
0121	Error when testing computer link	Replace the robot computer board if computer link is to be used

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
0122	Error when testing disk	a Check main cabling b Replace mass memory c Replace the robot computer board
0124	Error when testing the safety board	a Check that 24 V I/O is there. b Replace the safety board DSQC 228
0125	Error when testing the Winchester memory	a Check main cabling b Check that the massmemory gets 12 V. c Replace the massmemory. d Replace the robot computer board
0130	Robot computer board pd bus error	Replace the robot computer board
0131	Error when testing the programming unit	a Perform a restart b Check main cabling especially the 24 V supply. c Replace the programming unit
0133	Error when testing the monitor	a Check main cabling b Replace monitor c Check system parameters
0134	Non standard programming unit	a Perform a restart b Check main cabling c Replace the programming unit
0138	Not allowed I/O computer interruption	Replace the robot computer board
0139	I/O computer not responding	Replace the robot computer board
0140	Computer bus error, I/O computer	Replace the robot computer board
0201	Error at servo computer instruction test	Replace the robot computer board
0202	PROM checksum error, servo computer	a Replace the EPROM b Replace the robot computer board
0205	Erroneous servo computer "stall timer"	Replace the robot computer board
0206	Erroneous servo computer "time out timer"	Replace the robot computer board
0207	Erroneous I/O computer "interval timer"	Replace the robot computer board
0208	I/O computer error when handling not allowed command	Replace the robot computer board

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
0209	Error in communication between servo and main computers	Replace the robot computer board
0210	Error at ending of servo computer first test sections	Replace the robot computer board
0211	Error at ending of servo computer last test sections	Replace the robot computer board
0216	Servo computer initiation error	Replace the robot computer board
0239	Servo computer not responding	Replace the robot computer board
0240	Data bus error, servo computer	Replace the robot computer board
0302	Checksum error, axis computer sin-arctan table	Replace the robot computer board
0315	Error when loading test program to the axis computer	Replace the robot computer board
0358	Start-up error, axis computer	Replace the robot computer board
0701	Status bit error, FAULTRES-N, of the drive system	Replace the robot computer board
0702	Address bit error, SA0-SA5, of the drive system	Replace the robot computer board
0722	Status POWER UP-N erroneous from drive system after reset.	a Replace rectifier b Check flat cable between robot computer board and drive system, c Replace the robot computer, d Replace drive unit rack
0723	Signal RUNNING erroneous from drive system after reset.	a Replace rectifier b Check flat cable between robot computer board and drive system, c Replace the robot computer d Replace drive unit rack
0724	Status AC-N erroneous from drive system after reset.	a Replace rectifier b Check flat cable between robot computer board and drive system, c Replace the robot computer, d Replace drive unit rack
0731	Signal FAULT-N erroneous	Is used at manufacturing check of robot computer board

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
0732	Signal RUNNING erroneous	Is used at manufacturing check of robot computer board
0733	Read data error, R3 SDO	Is used at manufacturing check of robot computer board
0734	Read data error, R3 SD1	Is used at manufacturing check of robot computer board
0741	Axis computer does not respond at drive unit test	Replace the robot computer board
0742	Too large AD converter offset error	a Replace the robot computer board
0743	Too large AD converter offset error	a Replace the robot computer board
0744	Too large offset error on any drive unit channel	a Replace the robot computer board b Replace the drive unit
0751	The axis computer does respond at offset compensation, drive units	Replace the robot computer board
0900	The axis computer does respond at first contact to the measurement system of the specified axes	a Perform a restart b Install the missing equipment c Check cabling between measurement system and robot computer. d Replace measurement board DSQC 234 or axis board DSQC 233. e Replace the robot computer board
0902	Communication error to measurement system.	a Replace serialmeasurement board. b Replace the robot computer board
0903	Not allowed offset error on the X-channel of measurement system.	Replace serialmeasurement board.
0904	Not allowed offset error on the X-channel of measurement system.	Replace serialmeasurement board.
0905	Too large difference between X and Y channels of measurement system.	Replace serialmeasurement board.
0906	Disruption in cabling, the loop is closed without passing measurement board no 2. Linearity error on the X channel of measurement system. (X ramp does not increase)	A Do a restart. B Install missing equipment C Check the cabling between the measurement board and the robot computer. D Exchange the identified serialmeasurement board. E Exchange robotcomputer board.

ERROR MESSAGE	CAUSES	RECOMMENDED CORRECTIVE ACTION
0907	Linearity error on the Y channel of measurement system. (Y ramp does not increase)	Replace serial measurement board.
0920	Communication error to measurement system DSQC 233	a Replace measurement board DSQC 233 b Replace the robot computer board
0921	Offset error on AD converter to measurement system DSQC 233	Replace measurement board DSQC 233
0922	Offset error on DA converter, ch. 1 to measurement system DSQC 233	Replace measurement board DSQC 233
0923	Offset error on DA converter, ch. 2 to measurement system DSQC 233	Replace measurement board DSQC 233
0924	Offset error on DA converter, ch. 3 to measurement system DSQC 233	Replace measurement board DSQC 233
0925	Offset error on DA converter, ch. 4 to measurement system DSQC 233	Replace measurement board DSQC 233
0926	Offset error on DA converter, ch. 5 to measurement system DSQC 233	Replace measurement board DSQC 233
0927	Offset error on DA converter, ch. 6 to measurement system DSQC 233	Replace measurement board DSQC 233
0928	Linearity error, ch. 1 of measurement system DSQC 233	Replace measurement board DSQC 233
0929	Linearity error, ch. 2, 4, 6 of measurement system DSQC 233	Replace measurement board DSQC 233
0930	Linearity error, ch. 3 and 5 of measurement system DSQC 233	Replace measurement board DSQC 233
0960	Error at "local ring controller loopback"	Replace the robot computer board
0961	Error at "ring controller loopback"	Is used at manufacturing check of robot computer board
0962	Error at "ring controller loopback"	Is used at manufacturing check of robot computer board

Then, the robot is to be initiated in the test mode by pressing both push buttons on the computer board front, and then letting the initiation button up a couple of seconds before letting up the test button.

This enables obtaining an error code in cases where the error message is not displayed on the programming unit. The displayed subcode is one of the subcodes explained in section 5.

The error lamp on the robot computer board front will blink repeatedly to indicate the error subcode in accordance with the following description. The blinking of the error lamp will be repeated until the system is reinitiated.

560 ERROR/ROBOT COMPUTER BOARD



Test of board DSQC 233, axis board:

The error printouts available are the same as for a normal start-up according to sections 4 and 5.

I.e.

570 ERROR/MEASUREMENT SYSTEM AXES 1-6 xxxx
 571 ERROR/MEASUREMENT SYSTEM AXIS 7 xxxx
 572 ERROR/MEASUREMENT SYSTEM AXES 8-12 xxxx
 573 ERROR/MEASUREMENT SYSTEM AXES 7-12 xxxx
 (xxxx = sub codes)

Test of board DSQC 236, drive unit:

The error code parameter XX indicates output/input in accordance with the table below:

XX	Function	XX	Function
01	Enable loop error	02	Type code error
03	Test sequence error	04	Axis computer does not respond
05	Drive unit missing		

Test of board DSQC 209, analog in/output board:

The error printout parameter XX indicates the output/input in accordance to the table below:

XX	Function	XX	Function
0	Board absent	1	CHI
2	CH2	3	CH3
4	CH4		

Test of board DSQC 223 , digital in/output board:

The error printout parameter XX indicates the output/input in accordance to the table below:

XX	Function	XX	Function
0	Board absent	1	CHI
2	CH2	3	CH3
4	CH4	5	CH5
6	CH6	7	CH7
8	CH8	9	CH9
10	CH10	11	CH11
12	CH12	13	CH13
14	CH14	15	CH15
16	CH16		

Test of board DSQC 224, combined digital and analog in/output board:

The error printout parameter XX indicates the output/input in accordance to the table below:

XX	Function	XX	Function
0	Board absent	1	CHI
2	CH2	3	CH3
4	CH4	5	CH5
6	CH6	7	CH7
8	CH8	9	CH9
10	CH10	11	CH11
12	CH12	13	CH13
14	CH14	15	CH15
16	CH16		
21	Analog CHI	22	Analog CH 2

Keyword

Chapter, section

Absolute measurement	4.2
Acceleration index calculation	11.8.3.1
Adaptivity	10
Adaptivity sensor interface	10.1
Additional system I/O at I/O MAP	4.12.3
ALIGN Fool, external axis, home pos)	7.7
Analog ports	4.11.2
Arc welding	12
ARG, modify an argument	8.5
AUTO DEF, manual tool	9.4
AUTO mode	2.3
AUTO, automatic search	6.4.3, 10.3.3
Automatic (menu)	7
Automatic restart	4.2.3
Autosearch (AW)	6.4.3, 10.3.3.3
Arc weldning	12
Arc weldning, system principles	12.2
Arc weldning, program structure	12.3
Arc weldning of rear axis member	11.6.1
AW REST, Arcweld Restart	7.8
BASE, aligne base	7.7
Base point	3.3.1.1
BASEP, Basepoint active	3.2.4.5
BASEP, definition	9.5
BLOCK, from/to RAM	9.2.2
BWD, backward execution	7.1-4
CALL a sub-program	5.6
Change of operation mode	2.3.4
CHANGE, manual tool	9.4
CIRCEL	3.7, 6.13
Circuit breaker	2.2
CLEAN, welding	10.2.4.1, 10.2.5.2
CLEAR	9.1
Close-down	4.1.1, 4.1.3
Closely spaced positions	11.5.7
COM	5.13
Commands from robot	4.8.3
Commands to robot	4.8.2
Common functions	12.6.4
Compability with earlier programs	11.5.2
Computer link	2.9, 4.8
CONTOUR, contour tracing	6.8, 10.5
Connection and definition of sensors	10.2
Control cabinet	2.1
Control panel	2.3
Coordinate systems	3.2, 5.11. 11.7.3.1

Keyword	Chapter, section
Coordinate systems for defining positions	3.2.1
Coordinate systems in program execution	3.2.2
Coordinate systems for manual movements	3.2.3
COPY INS	8.16
COPY, external axes	8.5.4
COPY, program	8.10
COORD	5.11
CORNER 1, 2 (ZONE)	6.3
COR VEC, adaptive function	6.9, 10.6
Corner talking	3.6, 11.5.6
Cornering deviations	3.6.1.4
Correction vector	6.9, 10.5
CORV ST, adaptive correction	7.6
Cutting and deburing of plastic, forming of holes using the CIRCLE function	11.6.3
Cutting and deburing of plastic lining for a car door	11.6.4
Delay of outputs	5.3
DELETE	8.6
DELETE (DISK)	9.2.4
Definition of ARM load parameters	11.8.4
Define a position with PATH zone	12.6.1
Defining of WRIST load.	11.8.5
Deviations	3.6.4
DHANDCHK	3.4.1, 6.6, 6.10, 6.14, 6.16
Diagnostic messages	15.5
Dialog via prog.unit	2.4.1
Digital ports	4.11.1
DIR. direction search	6.4.2, 10.3.2
DISK	9.2
DISK (DELETE)	9.2.4
DISK (FROM DISK)	9.2.1
DISKdNTT)	9.2.3
DISK (TO DISK)	9.2.2
Diskette handling	4.7
Distance searching	10.3.1
DISPL, displacement of position	7.5
DISPL, modify a position	8.5.3
DIST, distance search	6.4.1, 10.3.1
Editing MENU	8
Editing	8.1
Emergency stop	1.2.1, 2.3, 2.5
END DATA, welding	12.5
Enabling device	2.6
ERASE, TCP, tool	9.4
Error buffer	4.6
ERASE	8.11
ERRORS	9.7, 9.12
Error buffer	15.2
Error hanling	13.4
Execution of instruction	14.3.6
Execution of instruction without current	14.3.7

Keyword	Chapter, section
Exceptions	3.6.1.5
External axes, homeposition	7.7.2
EXTAX, Independent External Axes	5.17
EXTC, welding	10.2.4.2
EXTPOS, welding	10.2.4.2
EXTFRAME	5.18, 6.18, 9.16, 12.6.5
FETCH, align fetch	7.7
FETCH, fetch from register	5.8.1
FINE, zone	3.6.1.2, 6.3
FLOW, gluing	13.1.1
FRDISK	9.2.1
FRSC	9.8
FRAME, reference frame	3.9.2, 5.15, 6.14, 9.6, 10.7
Function parameters	14.2.3, 14.3.2
GET B/ADD BLOCK	5.12
GLU SCA	5.19
GLUING	13.1
Gluing instruction	13.2.1
Gluing of a car door	11.6.2
GRIPPERS	5.1
HANDCHK	8.5.1
High performance programming	11.7
HOLD, HOLD RESET and AUTO INPUT	4.12.3.1
Hold-to-run	1.2.5
Home position	7.7.3, 9.17
Host computer	2.8, 4.8, 5.14
I/O signals	14.3.3
IN/OUT	9.4
Information(program)	4.5
INIT (DISK)	9.2.3
Initialize diskette	4.7.1
Initialization sequence	11.2
INPUT, manual	9.4
INPUT, wait-input	5.2.2
Input, jump-input	5.4.2
Inputs and outputs	4.12, 9.4
INS, instruction start	7.1-4
INSERT	8.7
INST NO, call up instruction	8.2
INST, modify instruction	8.5
INTER, interrupt	4.12, 5.10
Intialization sequence for the main program	11.2

Keyword	Chapter, section
Joystick	2.5
JUMP	5.4
JUMP (SEARCH)	5.4.3
JUMP(NINPOS)	5.4.4
LANG, language	9.11
Language section	4.3
Limitations	3.3.2.4
LIST	9.7
LIST, program printout	9.7
LOAD	9.5
LOAD, programmable	11.8
LOAD definition	9.5
Local (operation)	4.8.6
Location (Registers)	4.9.2
LOCATE	5.22
Logical Instruction menu	5
MAIN DATA, welding	12.4
Main Switch	2.2
MAN.DEF, manual tool	9.5
Manual menu	9.1
MANUAL REDUCED SPEED mode	2.3.2
MANUAL FULL SPEED mode	2.3.3
Manual restart	4.2.2
MAX, velocity	5.5
Manipulator	3.1
Manipulator, instructions	12.5.2, 12.6.3
Material handling applications	11.6.5
Measures to deal with overshoots	11.8.3.4
Menu overview	11.1
MIRROR	8.13
MODEXT	8.5.4
MODIFY	8.5
MODIFandMODARG	8.5.2
MODPOS	8.4
MODRECT COORD, modifiedrectangular coordinates	3.9, 5.11
Monitor	2.7
Moving the robot with the joystick	3.2.4
MOV REST	7.10, 4.12, 3.2
Movement characteristics	3.6
Movement principle	11.7.3.2
Movement velocity	3.5
N (not ready)	4.1.3
NINPOS, jump-not in position	5.4.4
NEXT, TCP, tool	9.5

Keyword	Chapter, section
Off-line	2.8
Off-line, teach	9.13
OPERATION	4
Operation status, computer link	4.8.6
Operation status, error list	15.3
OUTPUT	4.10, 5.3, 9.4
Override, (Gluing)	13.3
Override on the scaling instruction	13.3.2
Override, function (Arc welding)	12.7.2
Overview of five programming menus	11.3
PALETT	6.16, 9.15
Pattern program	11.4
Path following (principles)	3.6.1.1
Path optimization	3.6.1
Parallel displacement	3.9
Ports	4.11
POSPOS	6.10
Positioning instruction menu	6
Position programming	3.4
Position registers, location	4.9.2
POSLOC	6.12
Positioning	6
Power on. power off synchronization	4.1.1
PROG NO	8.12
PROGST.INS ST., BWD., SIM	7.1-4
PROG, printout	2.8
PROG.	8.8
Parallel displacement with reference point (REFP)	3.9.1
Preset values	11.8.3.3
Process application type plastic cutting, metal cleaning, deburring and polishing.	11.5.5
Program displacement with reference frame (FRAME)	3.9.2
Programming the first spot weld instruction	14.2.5
Programing robot path	11.5
Programing the spot weld instruction	14.3.5
Programing examples	11.6, 12.8
Program information	4.5
Programming unit	2.4
Program execution	4.1.2
Program execution with fixed TCP	3.3.2.2
Program information	4.5
Program printout	2.8

Keyword	Chapter, section
Program structure	11.3
Program test	12.7.1
Program robot paths	11.5
Programming unit	2.4
Position programming	3.4
RB MODE	9.10
Recommendations (for programming)	11.5.1-6
Recommended paths, zones	11.5.3-6
RECT COORD Rectangular base Coordinates system	3.2.4, 3.2.4.1, 3.2.4.4, 3.2.4.5
Recommendation for Arc Welding	11.5.3
Recommendation for Assembly, Machining, Tending and Material Handling.	11.5.6
Recommendation fo best performance	11.7.2
Recommendation for Glueing and Sealing	11.5.4
Request for superior control	4.8.5
REFP	3.9.1, 6.6
REG, registers	4.9, 5.8, 10.2.1
REG (FETCH)	5.8.1
REG (LOC)	5.8.4
REG (SET)	5.8.2
REG (TRANSFER)	5.8.3
RELTOOL	6.17
Remote control	2.10
Remote controlpanel signals, panel- I/O	4.13
RESEQ	8.9
RESYNK	7.9
Restart from PLC	4.2.4
Restart, after power failure	4.2
RETURN, from a program	5.7
ROBOT COORD, robot coordinates	3.2.4, 3.2.4.2, 3.2.4.3
Robot configuration	3.4.1
Room fixed TCP	3.3.2
Safety	1
SAME, repeat instruction	6.2
Save on disk	11.8.6
SC	2.9, 4.8, 5.14, 9.8, 9.9, 9.10
Scaling instruction	13.2.2
SEARCH	6.4, 10.3
SEARCH (auto)	6.4.3
SEARCH (dist)	6.4.1
SEARCH (dir)	6.4.2
SEARCH AUTO	6.4.3
Sensor definition	9.5
SIM, simulation of input	7.4
Signals	13.1.2
Singular Points	3.10
Smooth wrist reorientation	11.5.1
Softness definition	9.5
SOFTS	5.16

Keyword**Chapter, section**

Soft servo	3.8
Software configuration	4.4, 9.12.1
Speed control	10.4
Speed optimization	3.6.2
Spontaneous status messages	4.8.4
Spot welding	14
Spot welding instruction	14.2.4, 14.3.4
Spot welding instruction, editing	14.2.7
Standard SYSTEM I/O, inputs	4.12.1
Standard SYSTEM I/O, outputs	4.12.2
START DATA, welding	12.4
START	4.1
STEP	8.3
STO POS, storage of position	6.11
Stop	4.1.3
STORE, align store	7.7
SUCTRL, superior computer	5.14
SWI-function	14.3
SYS PAR	9.7.3
System-I/O	4.12
System faults	15.4
TCP	3.3
TCP, activate	5.9
TCP, fixed	3.2.4.6, 3.10
TCP, definition	9.5
TEACH, offline	9.13
Trimming of position gain index	11.8.3.2
TIME	6.15, 8.15
TIME, wait-time	5.2.1
TO SC program to SC	9.9
TO DISK	9.2.2
TOOL, (TCP, sensor, softs)	9.5
Tool direction	7.7.1
TRANSFER, register value to port	5.8.3
TRIM, programmable	11.9
USER PAR	9.7.4
V (%), percentage velocity	6.1, 8.1
VCTRL, speed adaptivity	6.7
VELOC	5.5
Velocity in circl	3.7.2
Velocity in corners	3.6.1.3
Velocity of reorient	3.6.2
Velocity in a zone	3.6.3
VISCTRL	5.22
WATT	5.2
WATT (TIME)	5.2.1
WAIT(INPUT)	5.2.2
WDATA, weld data	9.14, 10.2.3
WEAVING	6.5
WEAWE DATA, welding	12.4
Weight (M) and distance (X)	11.8.2

Keyword	Chapter, section
WELD	10.2.4.1
Welding data, description and programming	12.4
Welding gun, manual operation	14.2.2
Welding process, instructions	12.5.1, 12.6.2
Welding process, supervision	12.7.3
WEND, welding	10.2.4.1
WINCH (DELETE)	9.3.4
WINCH (FR WINCH)	9.3.1
WINCH (ESTII)	9.3.3
WINCH (TO WINCH)	9.3.2
WINSCHESTER MEMORY	9.3
WRIST/LOAD	5.20, 9.5, 11.8
Zones	3.6.1.2, 6.3