**FANUC** **R-30iB**

PROGRAMMING

MANUAL

Designed to accompany R-30iB

training courses within FANUC Nordic AB



*V 7.4*

**INTRODUCTION**

**Fanuc** R-30iB Training Course

This manual is intended to accompany the R-30iB training course

held within FANUC Nordic AB.

It is not intended as a comprehensive manual of all that a robot can do.

Please pay particular attention to the safety advice contained within this document.

For further reference please consult the comprehensive FANUC Reference Manual.

**Using This Manual**

Each section is individual labelled to enable you to quickly access the information you

require in addition to the table of contents on page 5.

**CONTENTS**

[**Safety**](#8)  ............................................... 6

[**The**](#18)**Basics** ............................................... 14

[Jogging](#18) [the](#18) [Robot](#18) [&](#18) [Coordinate](#18) Systems ............................................... [14](#18)

[Creating](#26) [a](#26) [Program](#26)  ............................................... [2](#26)0

[Testing](#30) [a](#30) [Robot](#30) Program ............................................... [23](#30)

Touchup ...............................................   [24](#34)

[Motion](#36) Default ...............................................   25

[Giving](#38) [2](#38) [Positions](#38) [the](#38) [Same](#38) [Identity](#38)  ............................................... 26

[**Program**](#40)**Elements**  ............................................... 27

[**EDCMD**](#48)[**Menu**](#48)  ............................................... 35

[**INST**](#58)[**Menu**](#58)  ............................................... 43

Registers ............................................... 43

[Position](#66) Registers ............................................... 46

[LPOS/JPOS](#70)  ............................................... 48

[**Inputs**](#74)[**and**](#74)[**Outputs**](#74)  ............................................... 49

[**IF/Select**](#80)**Statement** ............................................... 53

[IF](#80) [Statement](#80)  ............................................... 53

[Select](#84) [Statement](#84)  ............................................... 55

[**Wait**](#88)**Instruction** ............................................... 57

[**Labels**](#92)[**and**](#92)[**Jump**](#92)[**Labels**](#92)  ............................................... 59

[**Call**](#96)**Instruction** ............................................... 61

[**Miscellaneous**](#100)**Menu** ............................................... 62

[RSR](#100) [Instruction](#100)  ............................................... 62

[User](#102) Alarms ............................................... 62

Timers ...............................................   65

[Override](#108) [Instructions](#108)  ............................................... 67

[Remark](#110)  ............................................... 68

[Message](#112) [Instruction](#112)  ............................................... 69

[Parameter](#114) [Instruction](#114)  ............................................... 70

[Maximum](#118) [Speed](#118) [Instruction](#118)   ............................................... 72

**CONTENTS**

[**Multiple**](#120)[**Control**](#120)[**Menu**](#120)  ............................................... 73

[Run](#120) [Command](#120)  ............................................... 73

[Semaphore](#122) [Instruction](#122)  ............................................... 74

[**Program**](#124)[**Control**](#124)[**Instruction**](#124)  ............................................... 75

[Halt](#124) Instruction ............................................... 75

[Abort](#126) [Instruction](#126)    ............................................... 76

[**Offsets**](#128)[**Menu**](#128)  ............................................... 77

[Offsets](#128) ............................................... 77

[Frames](#132) Instruction  ............................................... 79

[**Skip**](#136)[**Condition**](#136)  ............................................... 81

[**Frames**](#138)**Overview** ............................................... 82

[Overview](#138)  ............................................... 82

[Tool](#140) [Frames](#140)  ............................................... 83

[User](#150) Frames ............................................... 89

[Jog](#154) [Frames](#154)  ............................................... 92

[**Utilities**](#158)  ............................................... 94

[Reference](#158) [Position](#158)  ............................................... 94

[Macros](#168)  ............................................... 96

[**Robot**](#158) **Payload Settings**  ............................................... 99

Setting a Schedule ............................................... 99

Setup of Payload Schedules ............................................... 101

[**File**](#200)[**Management**](#200)  ...............................................  102

[Program](#200) [Header](#200) Information ............................................... 102

[Deleting](#204) [a](#204) [Program](#204)  ............................................... 103

[Copying](#206) [a](#206) [Program](#206)  ............................................... 104

[**Backing**](#208)[**Up**](#208)  ............................................... 104

Loading and Restoring ............................................... 105

Image Backup  ............................................... 106

Restoring an Image Backup ............................................... 107

**SAFETY**

*Essential safety measures required whilst using robots*

Ensure you read and understand the FANUC Robotics SAFETY HANDBOOK.

**Teaching and manual operation**

* Don't operate the teach pendant and operator's panel with the gloves on your hand.
* Use a low override speed to increase your control over the robot when jogging it.
* Visualize the movement the robot will make before you press the jog keys on the teach pendant.
* Know the path that can be used to escape from a moving robot; make sure the escape path is never blocked.
* The area near the robot must be clean and free of oil, water, or debris.

**Production operation**

* Understand the complete task the robot is programmed to perform before

initiating production operation.

* Know the location and status of all switches, sensors, and control signals that

could cause the robot to move.

* Know where the EMERGENCY STOP buttons are located on both the robot

control and external control devices. Be prepared to press these buttons in an

emergency.

* Never assume that a program is complete if the robot is not moving. The robot

could be waiting for an input signal that will permit it to continue activity.

***FANUC Nordic AB*** *is not and does not represent itself as an expert in safety systems, safety equipment, or the specific safety aspects of your company and/or its work force. It is the responsibility of the owner, employer, or user to take all necessary steps to guarantee the safety of all personnel in the workplace.*

The appropriate level of safety for your application and installation can best be determined by safety system professionals.

FANUC Nordic AB therefore, recommends that each customer consult with such

professionals in order to provide a workplace that allows for the safe application,

use, and operation of FANUC Robot systems.

Additionally, as the owner, employer, or user of a robotic system, it is your responsibility

to arrange for the training of the operator of a robot system to recognize and respond to known hazards associated with your robotic system and to be aware of the recommended operating procedures for your particular application and robot installation.

FANUC Nordic AB therefore, requires that all personnel who intend to operate, program, repair, or otherwise use the robotics system be trained in an approved FANUC Robotics training course and become familiar with the proper operation of the system. Persons responsible for programming the system - including the design, implementation, and debugging of application programs - must be familiar with the recommended

programming procedures for your application and robot installation.

The following guidelines are provided to emphasize the importance of safety

in the workplace.

***SAFETY*** *is essential whenever robots are used*

**Safety Considerations**

Keep in mind the following factors with regard to safety:

* The safety of people and equipment
* Use of safety enhancing devices
* Techniques for safe teaching and manual operation of the robot(s)
* Techniques for safe automatic operation of the robot(s)
* Regular scheduled inspection of the robot and workcell
* Proper maintenance of the robot

The safety of people is always of primary importance in any situation.

However equipment must be kept safe too.

When prioritizing how to apply safety to your robotic system, consider the following:

* People
* External devices
* Robot(s)
* Tooling
* Workpiece

Always give appropriate attention to the work area that surrounds the robot.

The safety of the work area can be enhanced by the installation of some or all of

the following devices:

* Safety fences, barriers, or chains
* Light curtains
* Interlocks
* Pressure mats
* Floor markings
* Warning lights
* Mechanical stops
* DEADMAN switches
* EMERGENCY STOP buttons

**A Safe Workcell**

A safe workcell is essential to protect people and equipment. Observe the following

guidelines to ensure that the workcell is set up safely.

* Sponsor your personnel for training in approved FANUC Robotics training

course(s). Never permit untrained personnel to operate the robots.

* Install a lockout device that uses an access code to prevent unauthorized

persons from operating the robot.

* Use anti-tie-down logic to prevent the operator from bypassing safety

measures.

* Install safety fences to protect against unauthorized entry by personnel into

the work envelope.

**A Safe Workcell**

* Arrange the workcell so the operator faces the workcell and can see what is

going on inside the cell.

* Clearly identify the work envelope of each robot in the system with floor

markings, signs, and special barriers. The work envelope is the area defined

by the maximum motion range of the robot, including any tooling attached to

the wrist flange that extends this range.

* Position all controllers outside the safety fence.
* Never rely on software as the primary safety element.
* Mount an adequate number of EMERGENCY STOP buttons or switches within

easy reach of the opera`tor and at critical points inside and around the outside

of the workcell.

* If necessary, install flashing lights and/or audible warning devices that activate

whenever the robot is operating, that is, whenever power is applied to the

servo drive system.

* Install special guarding that prevents the operator from reaching into restricted

areas of the work envelope.

* Use interlocks.
* Use presence or proximity sensing devices such as light curtains, mats, and

capacitance and vision systems to enhance safety.

* Periodically check the safety joints or safety clutches that can be optionally

installed between the robot wrist flange and tooling. If the tooling strikes an

object, these devices dislodge, remove power from the system, and help to

minimize damage to the tooling and robot.

* Make sure all external devices are properly filtered, grounded, shielded, and

suppressed to prevent hazardous motion due to the effects of electro-

magnetic interference (EMI), radio frequency interference (RFI), and electro-

static discharge (ESD).

* Eliminate pinch points. Pinch points are areas where personnel could get

trapped between a moving robot and other equipment.

* Provide enough room inside the workcell to permit personnel to teach the

robot and perform maintenance safely.

* Program the robot to load and unload material safely.
* If high voltage electrostatics are present, be sure to provide appropriate

interlocks, warning, and beacons.

* If materials are being applied at dangerously high pressure, provide electrical

interlocks for lockout of material flow and pressure.

**Observe the following rules**

* Advise all personnel who must teach the robot or otherwise manually operate

the robot to observe the following rules:

* Know whether or not you are using an intrinsically safe teach pendant if you

are working in a hazardous environment.

* Before teaching, visually inspect the robot and work envelope to make sure

that no potentially hazardous conditions exist. The work envelope is the area

defined by the maximum motion range of the robot. These include tooling

attached to the wrist flange that extends this range.

* The area near the robot must be clean and free of oil, water, or debris.

Immediately report unsafe working conditions to the supervisor or safety

department.

* FANUC Robotics (UK) Ltd. recommends that no one enter the work envelope

of a robot that is on. However, if you must enter the work envelope, be sure

all safeguards are in place, and check the teach pendant DEADMAN switch

for proper operation. Take the teach pendant with you, turn it on, and be

prepared to release the DEADMAN switch. Only the person with the teach

pendant should be in the work envelope.

* Know the path that can be used to escape from a moving robot; make sure the

escape path is never blocked.

* Isolate the robot from all remote control signals that can cause motion while

data is being taught.

**Testing a Program**

Test any program being run for the first time in the following manner:

* Using a low motion speed, single step the program for at least one full cycle.
* Using a low motion speed, test run the program continuously for at least one

full cycle.

* As speed is increased, the path may vary slightly. Run through the program

at 5-10% intervals up to 100%.

* Using the programmed speed, test run the program continuously for at least

one full cycle.

**Running Production**

Make sure all personnel are outside the safety fence before running production.

Advise all personnel who operate the robot during production to observe the following

rules:

* Know the entire workcell area. The workcell includes the robot and its work

envelope, plus the area occupied by all external devices and other equipment

with which the robot interacts.

* Understand the complete task the robot is programmed to perform before

initiating production operation.

* Make sure all personnel are outside the safety fence before operating the

robot.

* Never enter or allow others to enter the work envelope during production

operation of the robot.

* Know the location and status of all switches, sensors, and control signals that

could cause the robot to move.

* Know where the EMERGENCY STOP buttons are located on both the robot

control and external control devices. Be prepared to press these buttons in an

emergency.

* Never assume that a program is complete if the robot is not moving. The

robot could be waiting for an input signal that will permit it to continue activity.

* If the robot is running in a pattern, do not assume it will continue to run in the

same pattern.

* Never try to stop the robot, or break its motion, with your body. The only way

to stop robot motion immediately is to press an EMERGENCY STOP button

located on the controller panel, teach pendant, or emergency stop stations

around the workcell.

**Inspecting the Robot**

When inspecting the robot, be sure to

* Turn off power at the controller.
* Lock out and tag out the power source at the controller according to the

policies of your plant.

* Turn off the compressed air source and relieve the air pressure.
* If robot motion is not needed for inspecting the electrical circuits, press the

EMERGENCY STOP button on the operator panel.

* If power is needed to check the robot motion or electrical circuits, be prepared

to press the EMERGENCY STOP button, in an emergency.

**Maintenance**

When performing maintenance on your robot system, observe the following rules:

* Never enter the work envelope while the robot or a program is in operation.
* Before entering the work envelope, visually inspect the workcell to make sure

no potentially hazardous conditions exist.

* Consider all or any overlapping work envelopes of adjoining robots when

standing in a work envelope.

* Test the teach pendant for proper operation before entering the work

envelope.

* If it is necessary for you to enter the robot work envelope while power is turned

on, you must be sure that you are in control of the robot. Be sure to take the

teach pendant with you, press the DEADMAN switch, and turn the teach

pendant on. Be prepared to release the DEADMAN switch to turn off servo

power to the robot immediately.

* Whenever possible, perform maintenance with the power turned off. Before

you open the controller front panel or enter the work envelope, turn off and

lock out the 3-phase power source at the controller.

* Release or block all stored energy. Before working on the pneumatic system,

shut off the system air supply and purge the air lines.

* Isolate the robot from all remote control signals. If maintenance must be done

when the power is on, make sure the person inside the work envelope has

sole control of the robot. The teach pendant must be held by this person.

* Make sure personnel cannot get trapped between the moving robot and other

equipment. Know the path that can be used to escape from a moving robot.

Make sure the escape route is never blocked.

* Use blocks, mechanical stops, and pins to prevent hazardous movement by

the robot. Make sure that such devices do not create pinch points that could

trap personnel.

* Be aware that when you remove a servomotor or brake, the associated axis

will fall if it is not supported or resting on a hard stop.

* When replacing or installing components, make sure dirt and debris do not

enter the system.

* Use only specified parts for replacement. To avoid fires and damage to parts

in the controller, never use non-specified fuses.

* Before restarting a robot, make sure no one is inside the safety fence; be sure

that the robot and all external devices are operating normally.

**Keeping Machine Tool & Other Devices Safe**

Certain programming and mechanical measures are useful in keeping the machine tools and other external devices safe. These measures are outlined below.

Implement the following programming safety measures to prevent damage to machine tools and other external devices.

* Back-check limit switches in the workcell to make sure they do not fail.
* Implement ``failure routines" in programs that will provide appropriate robot

actions if an external device or another robot in the workcell fails.

* Use handshaking protocol to synchronize robot and external device

operations.

* Program the robot to check the condition of all external devices during an

operating cycle.

* Implement the following mechanical safety measures to prevent damage to

machine tools and other external devices.

* Make sure the workcell is clean and free of oil, water, and debris.
* Use software limits, limit switches, and mechanical hardstops to prevent

undesired movement of the robot into the work area of machine tools and

external devices.

**Preventing Damage During Operation**

Observe the following operating and programming guidelines to prevent damage

to the robot.

* The following measures are designed to prevent damage to the robot during

operation.

* Use a low override speed to increase your control over the robot when jogging

the robot.

* Visualize the movement the robot will make before you press the jog keys on

the teach pendant.

* Make sure the work envelope is clean and free of oil, water, or debris.
* Use circuit breakers to guard against electrical overload.

**Preventing Damage During Programming**

The following safety measures are designed to prevent damage to the robot during programming:

* Establish interference zones to prevent collisions when two or more robots

share a work area.

* Make sure that the program ends with the robot near or at the home position.
* The robot must be at HOME to start a cycle, therefore the robot must return

HOME at the end of the program.

* Be aware of signals or other operations that could trigger operation of tooling

resulting in personal injury or equipment damage.

***NOTE****: Any deviation from the methods and safety practices described in this manual must conform to the approved standards of your company. If you have questions, see your supervisor.*

**BASICS**

JOGGING THE ROBOT

*The process of moving the robot using jog keys on the teach pendant*

***Remember!*** *Only the person with the teach pendant should be in the working envelope of the robot*

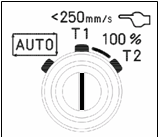
Jogging the robot is the process of moving it around.

*It is* ***ESSENTIAL*** *that all safety advice is understood and followed during jogging of the robot.*

Always begin jogging the robot at a LOW speed. (T1 mode)

Become familiar with the different ways the robot moves depending on the different

coordinate system chosen

******Mode Selection Switch (keyswitch on controller front)**

**Auto**

1. The robot can be operated at maximum speed.
2. The safeguards are active.

**T1 (Setup mode for testing programs)**

1. Robot motion can be triggered and started only from the teach pendant.
2. All speeds are limited not to exceed 250 mm/sec.
3. It is possible to operate the robot only when the deadman switch is pressed.
4. The FENCE safeguard is not active.

**T2 (Testing at operation speed)**

1. Robot motion can be triggered and started only from the teach pendant.
2. The robot can be operated at maximum speed.
3. It is possible to operate the robot only when the deadman switch is pressed.
4. The FENCE safeguard is not active.

Note:

If the mode is changed from Auto to another mode, the program will be stopped.

**Coordinate Systems**

The robot may be jogged in one of four different coordinate systems.

Each system moves the robot in a different way.

**World Frame**

The default system of the robot for XYZ moves is the WORLD frame.

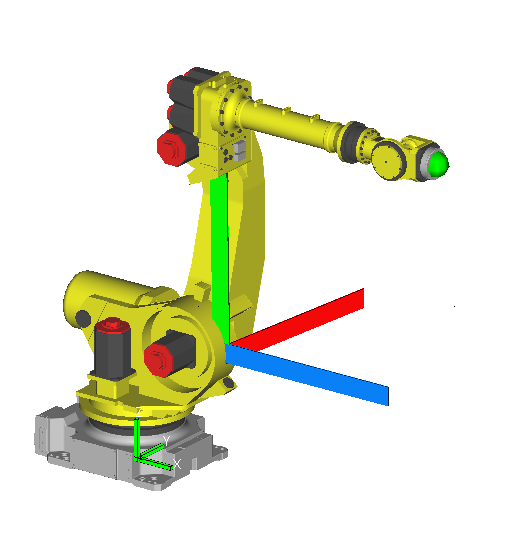
WORLD is a cartesian (XYZ) frame. That is, it moves the robot in a frame where the

X, Y and Z directions are at right angles to each other.

In WORLD, multiple axes at a time move to keep the robot Tool Center Point

in an XYZ plane.

WORLD FRAME



Z

X

Y

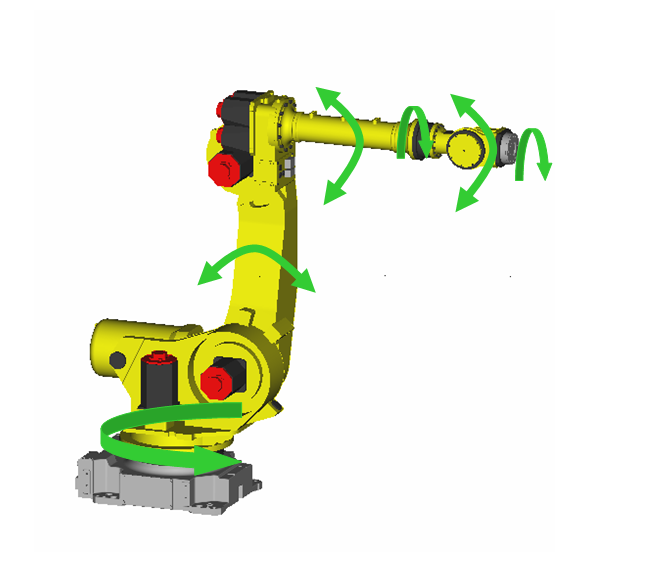
**Joint Coordinate System**

Joint Coordinate system allows you to move only one axis at a time.

That is, one joint at a time unless multiple jog keys are pressed.

The zero point of this system is on each individual axis.

JOINT is the default system of the robot and cannot be changed.

 Joint Coordinate system

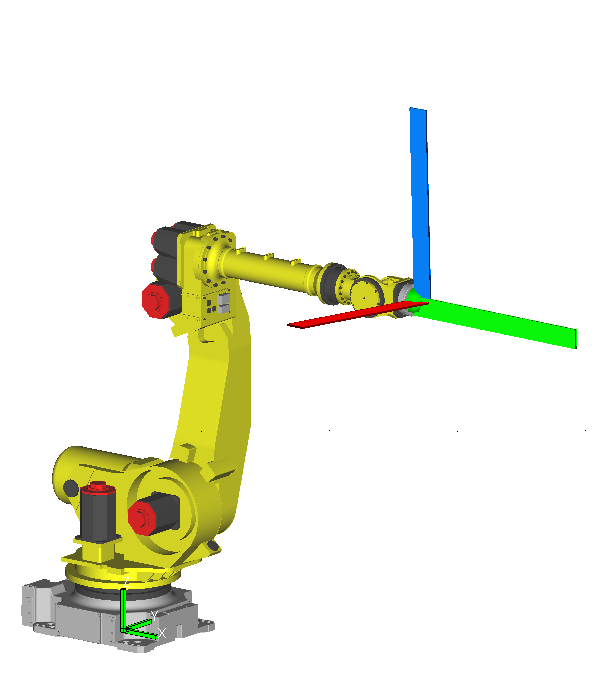
**Tool Frame**

Moves multiple axes at a time relative to the robot faceplate.

The robot will move relative to the tool tip if a Tool Center Point has been set up.

The zero point of the robot is on the faceplate but this can be moved to the tip

of the tool which is known as the Tool Center Point (TCP)

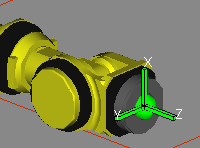


[Tool](#21) [Coordinate](#21) [System](#21)

X

Y

Z

Axis 4, axis 5 and axis 6 are in the ZERO degree position

**User Coordinate System**

User is a cartesian system, or XYZ.

Moves multiple axes at a time to ensure that the Tool Center Point or robot

faceplate moves in the selected XYZ plane.

The zero point of the robot can be placed anywhere by setting up a user frame.

If a user frame has not been set up, the robot will move in the WORLD frame

when user coordinate system is selected.

***CAUTION…*** *When a user frame is set up it can affect how your programs work.*

*CONSULT THE SECTION ON USER FRAMES.*

**Jog Frame (JGFRM) Coordinate System**

Jgfrm is a cartesian system, or XYZ.

Moves multiple axes at a time to ensure that the Tool Center Point or robot faceplate

moves in the selected XYZ plane.

A jog frame can be set up by the user to allow for easier jogging in an XYZ plane

other than the default.

The robot zero point of a jog frame always remains in the WORLD frame.

If a JOG frame has not been set up, the robot will move in the WORLD frame

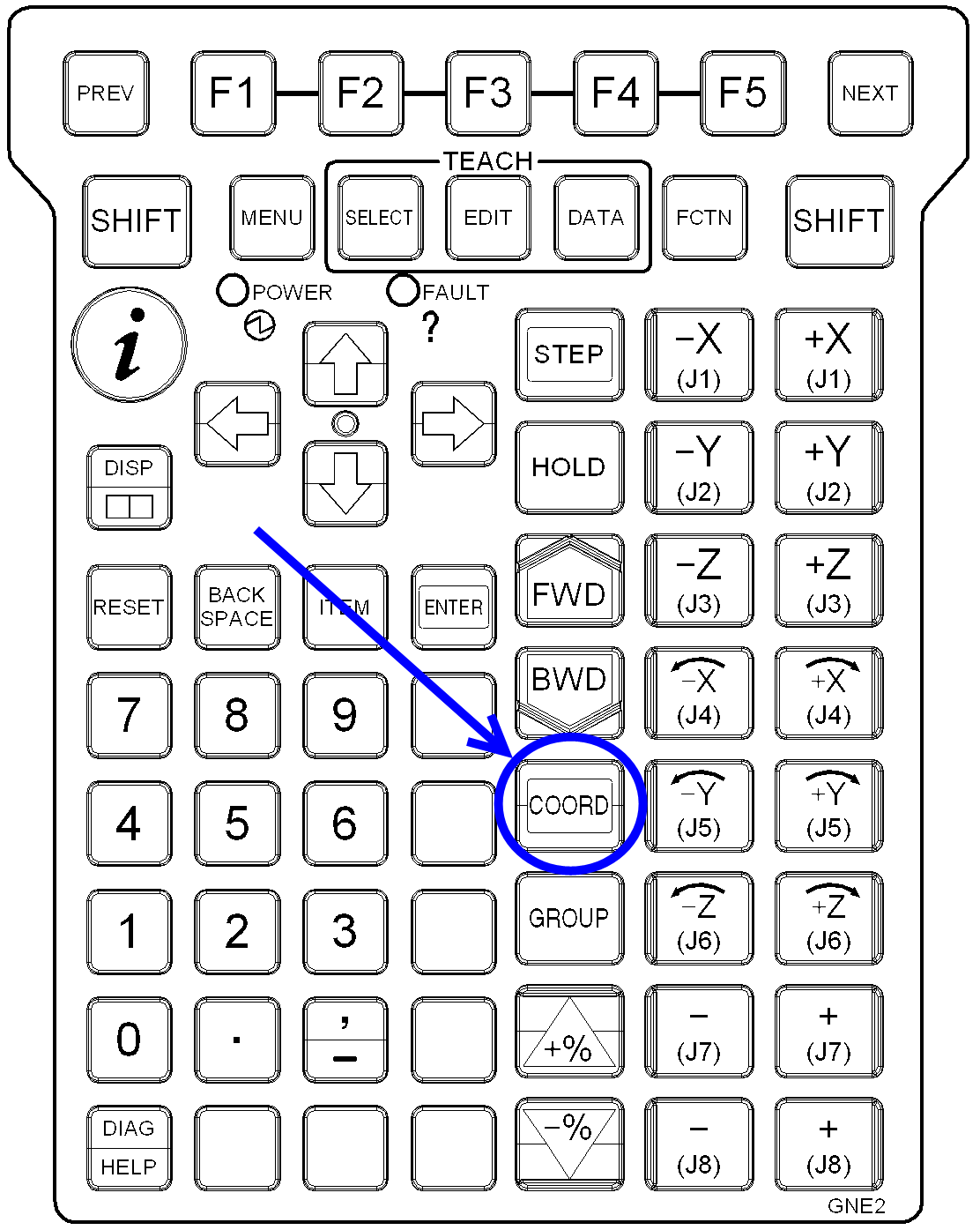
when JGFRM coordinate system is selected.

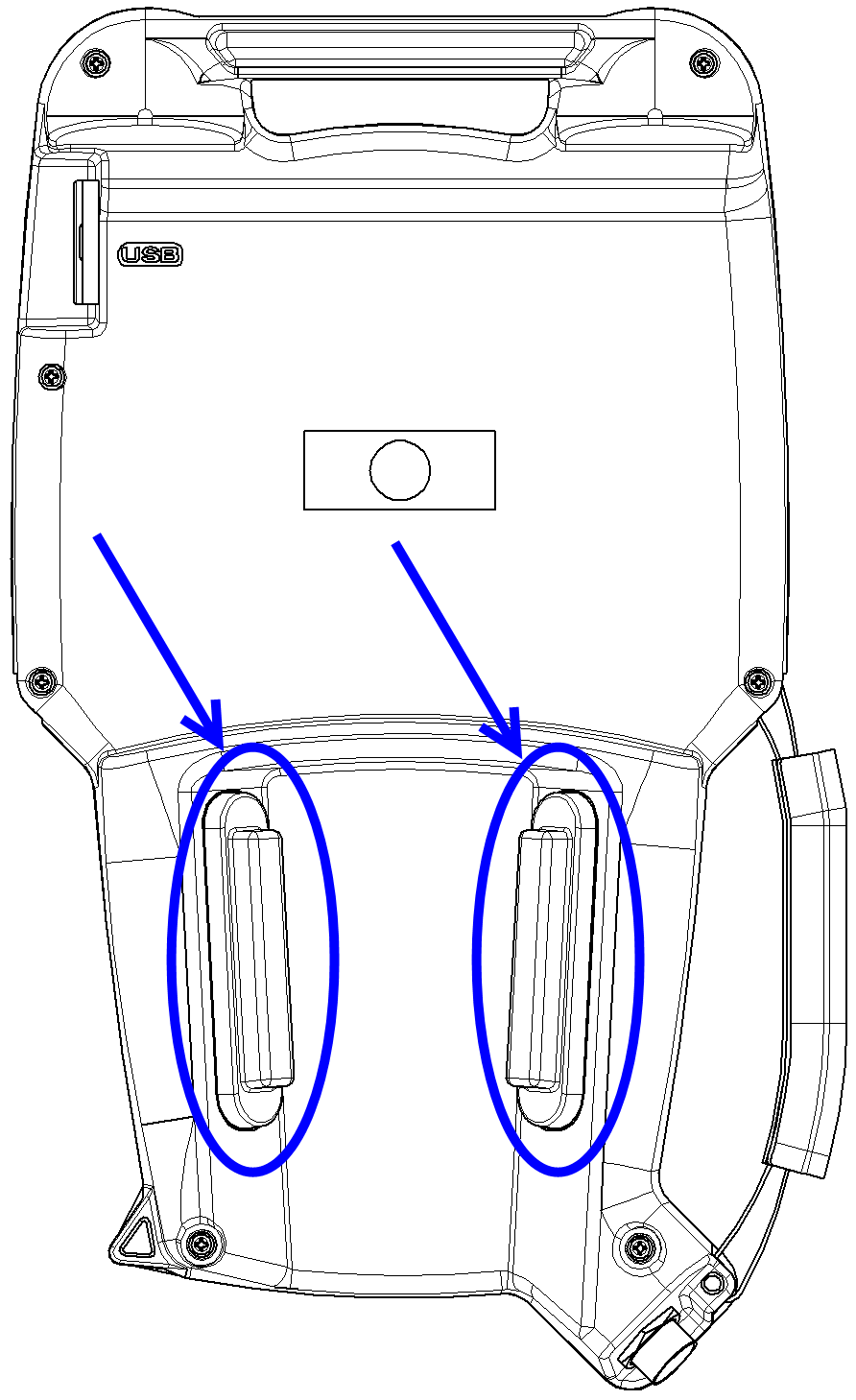
A jog frame has no effect on a program

Consult the Section on Jog Frames.

**Activating Coordinate Systems**

Use the COORD Key  to change between Coordinate systems.





**Jogging the Robot**

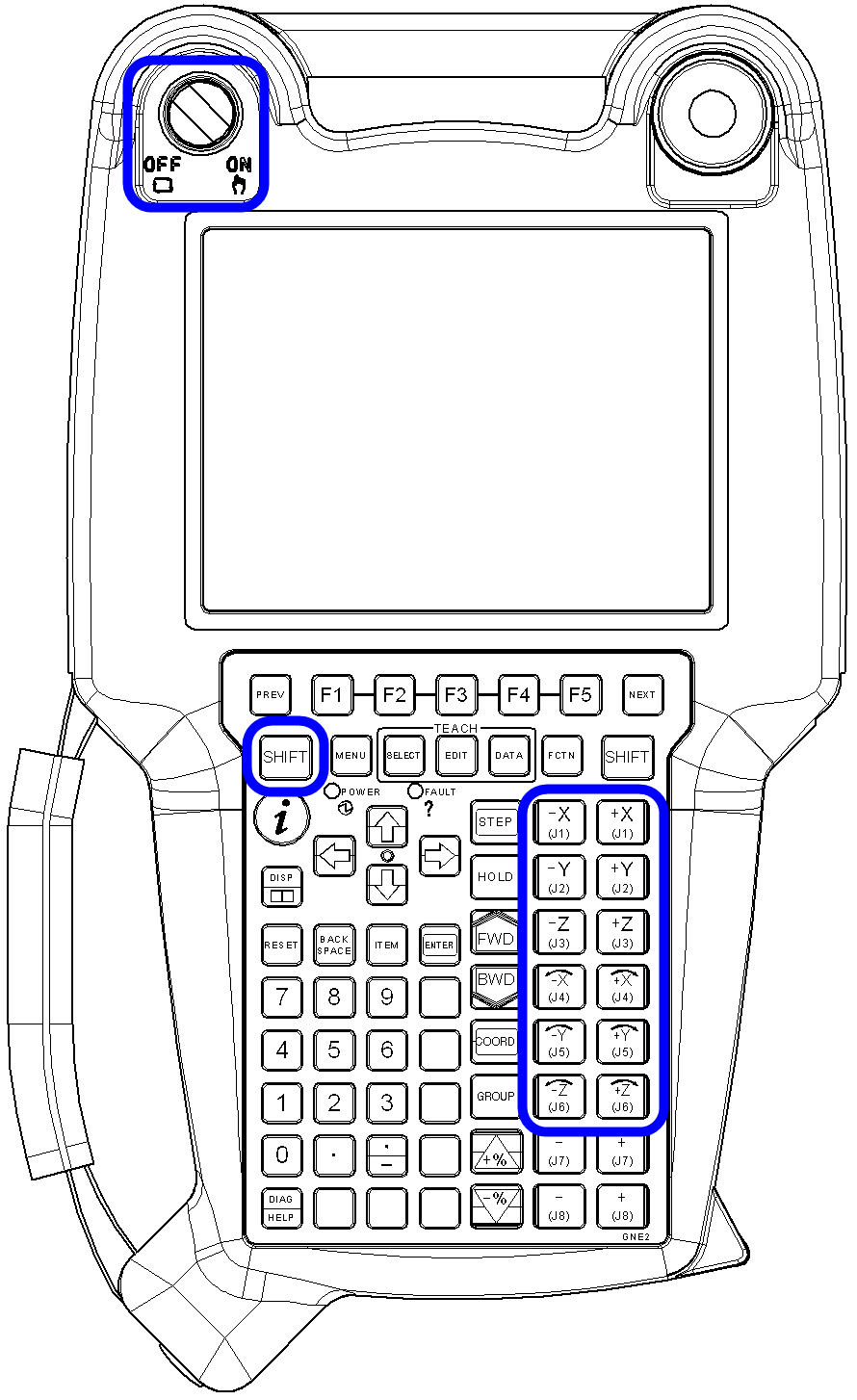
***CAUTION****: The following steps cause the robot to move, take care!*

1. To jog the robot, press and hold the Deadman

Handle on the rear of the teach pendant.

1. Turn the teach pendant ON/OFF switch to ON, and

if necessary press the RESET key to reset any faults



Two keys are needed to  jog (move) the robot.

One of the two SHIFT keys and one of the 12 jog keys.

1. Press and hold any one of the SHIFT keys
2. Press one of the twelve jog keys to the

right of the teach pendant 🡪

1. The robot will now move in whichever

coordinate system is active.

**CREATING A PROGRAM**

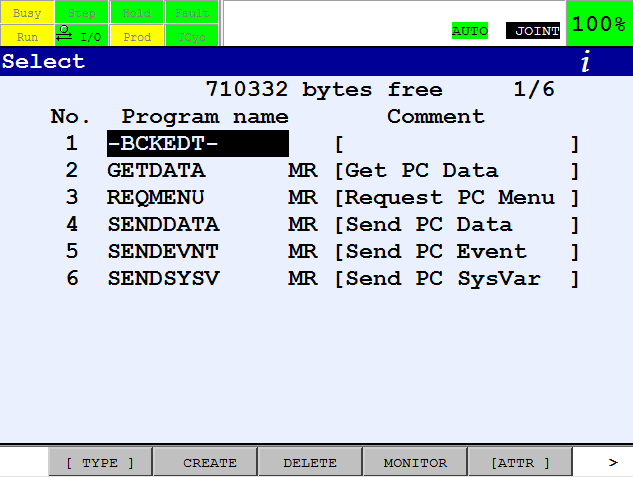
*Runs step by step through the procedure for creating a new program*

Before you can begin to teach the robot the path to take around an object, you must

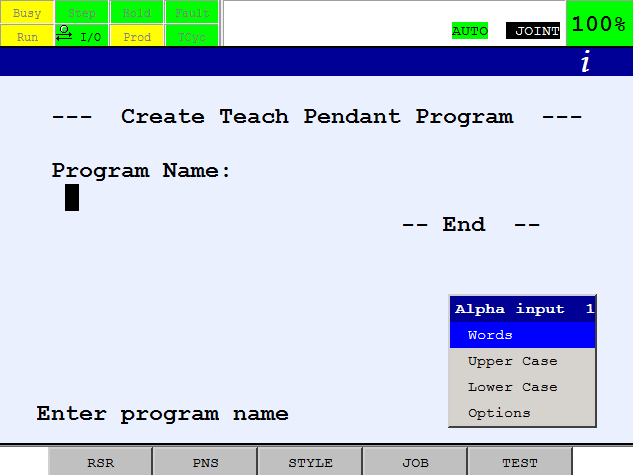
first create a program. You must give it a unique program name to identify it, and this

selection illustrates how this is achieved.

**Creating a Program**



1. Pick up the teach pendant
2. Press the SELECT Key
3. The SELECT screen will be displayed



1. If CREATE is not above F2,
2. press NEXT key
3. Press F2, CREATE
4. The following screen

should be displayed 🡪



1. Press the key to Uppercase and all the

letters of the alphabet should be above



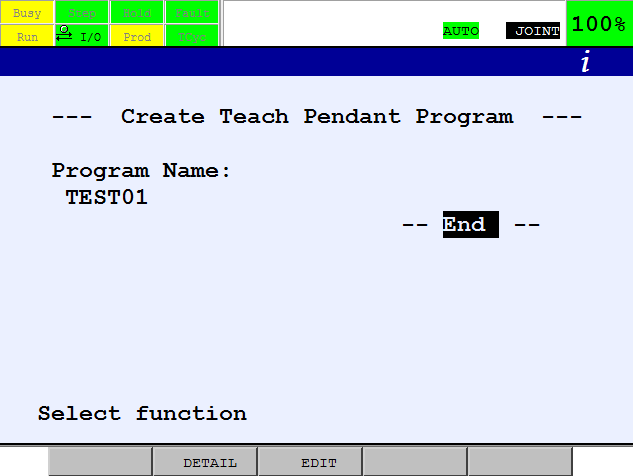
the function keys, like so:

1. To select a letter, press the corresponding function key (F1-F5) until the required letter

appears in the program name box.



*NOTE: If 2 consecutive letters in the program name appear above the same function key, press the right cursor key*



1. Continue to press the function keys

corresponding to the required letters

until the program name is complete.



1. If you make a mistake,

use the BACKSPACE key.

1. Once the program name is complete,

press ENTER

1. The cursor will jump to – END –
2. Press ENTER again to lead to the



Program screen.

**Recording a Position**

The teach pendant is now ready for the first program point to be recorded.

1. To do this, jog the robot to the first position and press SHIFT + [F1] (POINT).
2. The robot has now been taught that position and a line identifying it will appear looking something like this:

J P[1] 100% FINE

1. Continue jogging the robot to new positions and recording

them by using SHIFT +F1, POINT.

Every time SHIFT + [F1], POINT are pressed, new program lines will be displayed

representing the robot positions in space.

1. The new program will look similar to this:

J P[1] 100% FINE

J P[2] 100% FINE

J P[3] 100% FINE

J P[4] 100% FINE

J P[5] 100% FINE

J P[6] 100% FINE

1. Each new recorded position is given the next available position number, P[ ].

**TESTING A ROBOT PROGRAM**

*Running through a robot program for the first time.*

*Ensure all safety advice has been read and understood. The following steps cause the robot to move, take care.*

1. To test a program for the first time, cursor to the top of the program and press the

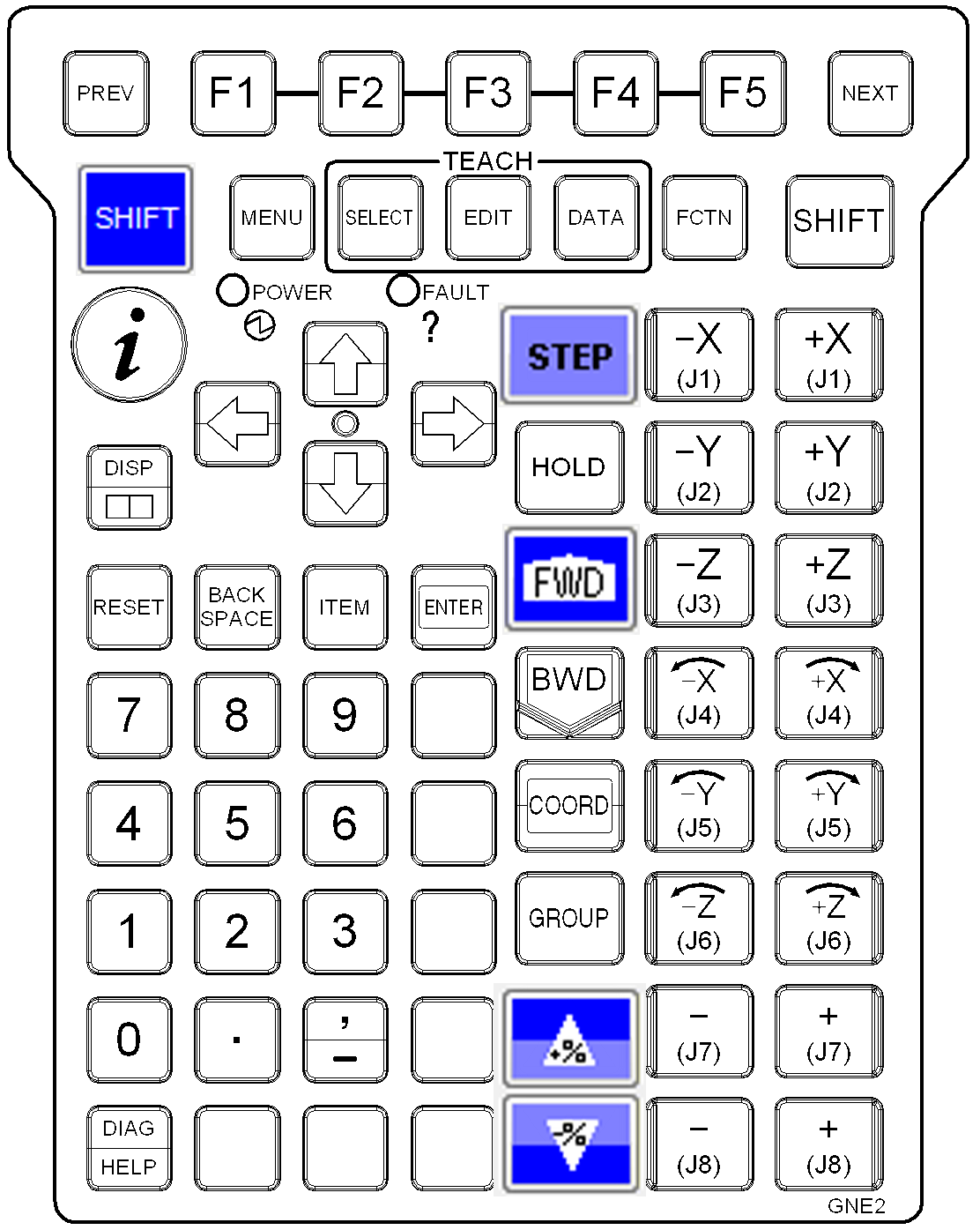




STEP key to activate the STEP function teach pendant screen.



1. Ensure a**SLOW** speed  is selected, using the SPEED Keys.



1. Press and hold the SHIFT key
2. Press the FWD key
3. So long as the STEP LED is on, the robot will

move to the first position and stop.

1. Keep hold of the SHIFT key and press the FWD

key repeatedly until the program

finishes execution

**To stop the robot at any time:**

* Release the SHIFT key
* Let go of the Deadmans
* Hit any emergency stop.

**Running Through the Program Continuously**

When you are happy that the program is running as it should, press STEP again.



The STEP indicator will turn green and the robot is in **continuous mode.**

1. Ensure a SLOW speed is selected

2. Press and hold the SHIFT key

3. Press the FWD key

The robot will run the program through completely.

Continue to run the program, increasing the percentage speed slowly, bearing in mind that the path of the robot may change the faster you go.

To re-check a program point, press SHIFT & BWD to move back one step at a time.

**TOUCHUP**

*Altering a program position*

The TOUCHUP command may be needed to modify the position of existing points.

It is located above F5.

**Using TOUCHUP**

To use Touchup, you must press SHIFT + TOUCHUP whilst in the program screen.

This is similar to recording a point using SHIFT + POINT.

The difference between the two methods of recording positions is:

**SHIFT + POINT**

Records a new position with the default set of program elements.

**SHIFT + TOUCHUP**

Changes the positional information only of the position the cursor is currently on*.*

**Set New I.D**

If two points in a program have the same identity.

for example if there are two P[3] ‘s, the teach pendant will ask if you want to set

a new i.d when you try to touchup one of them.

Set New I.D.

YES NO

F4 F5

If F4 YES is selected, the position is touched up and given the next available position

number. The second position in the program will not be touched up and the two

no longer have the same identity.

If F5 NO is selected, the position is touched up and the second position is

touched up also, keeping the two positions identical.

**MOTION DEFAULT**

*Changing the way a position is recorded as default*

The usual program line that appears when SHIFT + POINT is pressed looks similar to:

J P[1] 100% FINE

It is possible to change what kind of program line appears when SHIFT + POINT is

pressed. That is, to change the default.

There are 4 possible motion defaults.

**Changing the Default Motion**



* 1. To select from the default

menu pressF1 POINT

(without the SHIFT key)

* 1. A screen similar to that

below will appear 🡪

* 1. Press F1, ED\_DEF, and these motion

defaults can be changed to that required by the user.

* 1. Just cursor in, press CHOICE (above F4) and edit these defaults.

When completed press DONE (F5).

**GIVING 2 POSITIONS THE SAME ID**

*Recording identical positions*

This procedure allows you to give a position the same identity as a previously recorded

one.

1. The program line will look like so:

J P[5] 100% FINE

1. To give P[5] the same identity as P[2] for example, cursor onto the 5 like so:

J P[\_] 100% FINE

1. Enter a 2 with the numeric keypad, and press ENTER.
2. The program line will now look like so:

J P[2] 100% FINE

1. T his point has now taken the same identity as a previously recorded position number [2]. If there is no position number [2] in the program, the position number you have changed will be uninitialized. Ensure you enter a number that exists in the program.

***NOTE****: This procedure can be carried out to give any position the same identity as a previously recorded one.*

**PROGRAM ELEMENTS**

**PROGRAM ELEMENTS**

*Altering the way the robot moves to positions*

A program line is made up of various elements that control how the robot moves between

two positions.

Each element can be changed to enable the robot to move the required path in the

desired way.

A typical program line will look like so:

**J P[ ] 100% FINE**

POSITION

MOTION

TERMINATION

SPEED

Each item in the program line is a **Program Element.**

Every element can be changed and each are described in this section.

***NOTE****: To change any element, cursor onto it and CHOICE appears above F4.*

*Pressing CHOICE will bring up various menus depending on which element the cursor is placed.*

There are further advanced options available by cursoring to the very end of the line, past

FINE and pressing F4 CHOICE.

**MOTION**

The motion type defines the way in which the robot moves between two points. To

change it, place the cursor on the first element, and select CHOICE (above F4).

J P[1] 100% FINE

The following options are available:

1. Joint

2. Linear

3. Circular

**Joint**



The robot makes its own way between two

points and calculates the fastest way to get

there. Joint motion is particularly useful for

points in space when it doesn’t matter which

path the robot takes.

Joint motion is programmed at the destination

position.

It causes the robot to move all the required

axes to the destination position

simultaneously.

Units of Joint motion can be a percentage of

the total default speed, or seconds.

The actual speed of a Joint move is

dependant on the speed of the slowest axis.

**Linear**

The robot will move the Tool Center Point between two positions in a straight line.

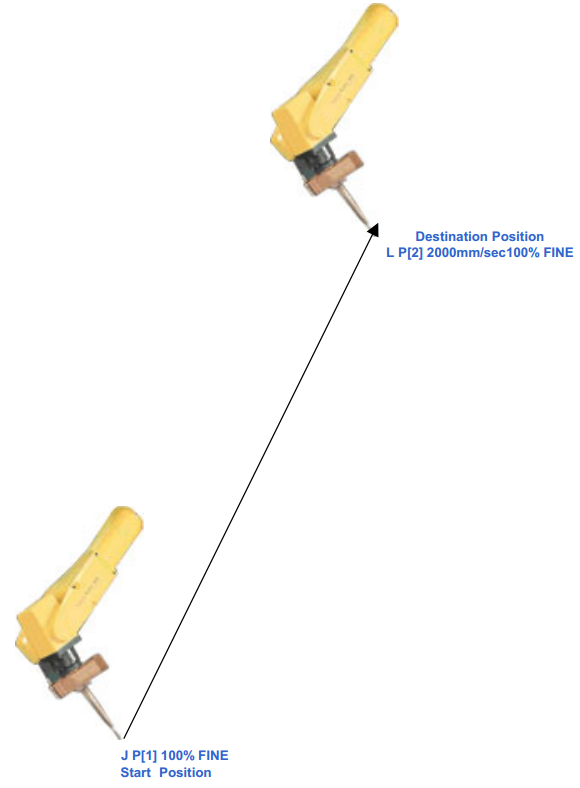
Linear motion is programmed at the destination position.

Units of linear motion can be millimetres per second, inches per minute,

centimetres per minute, degrees per second and seconds.

The orientation of the Tool Center Point changes gradually as the robot moves

from the start position to the destination position.



**Circular**

With circular motion selected, the robot will move the Tool Center Point in an arc from the

start position through an intermediate position to the end position.

Circular motion is programmed at the intermediate position.

Units of circular motion are millimetres per second, inches per minute, centimetres per

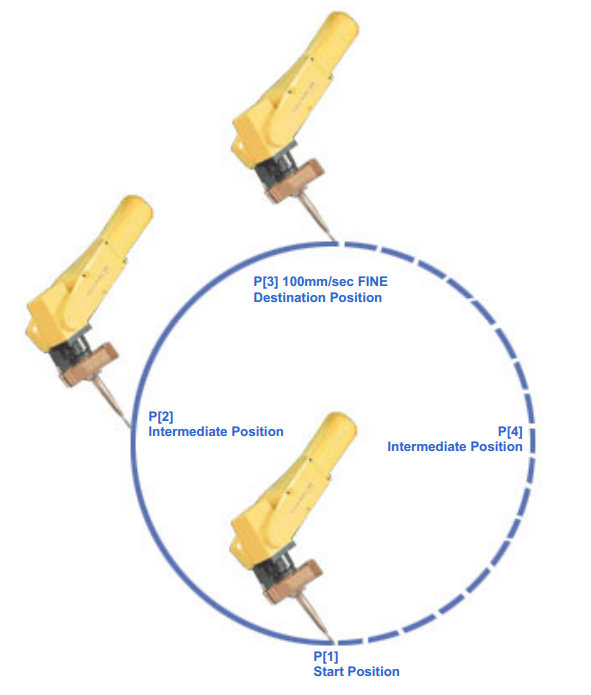
minute, degrees per second and seconds.

Program Example:

**J** P [1] 100% FINE (start point)

**C** P [2] (passing point)

P [3] 2000 mm/sec FINE (end point)



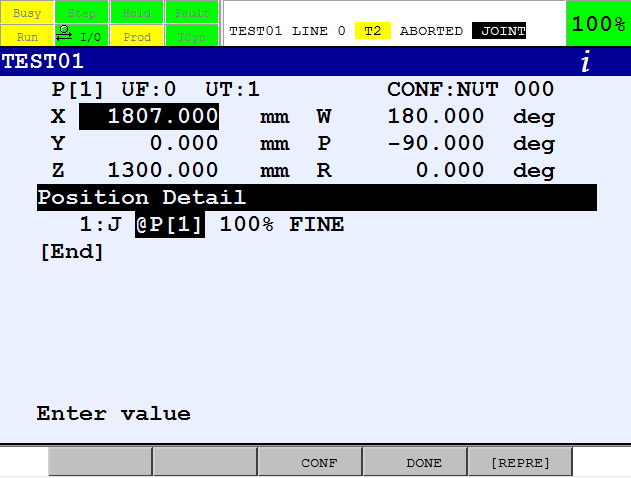
**Position**

J P[1] 100% FINE

Place the cursor on any position number to examine the positional coordinates of that

position.

Press F5 POSITION:



The screen displayed informs you where in space this position was recorded. You can

edit all the values by cursoring onto them and using the numeric keypad to enter new

values.

X,Y,Z,W,P and R are the cartesian coordinates of the selected position.

To display the JOINT coordinates (each axis in degrees), press F5 [REPRE] and select

Joint. When coordinates are displayed in JOINT, all tool and user frame information is

ignored.

**Speed**

J P[1] 100% FINE

The speed value can be changed by cursoring onto it and entering a new value via the

numeric keypad.

The speed units can be changed by cursoring onto them and pressing F4 CHOICE.

The choice available depends on the motion type selected, Joint, Linear or Circular.

If the motion is JOINT the following units are available for selection:

1. %

2. secs

3. msec

If the motion is LINEAR or CIRCULAR the following units are available for selection:

1. mm/sec 5. sec

2. cm/min 6. msec

3. inch/min 7.

4. deg/sec 8.

The robot knows how fast it can go and will not let you enter too fast a speed.

**Termination**

J P[1] 100% FINE

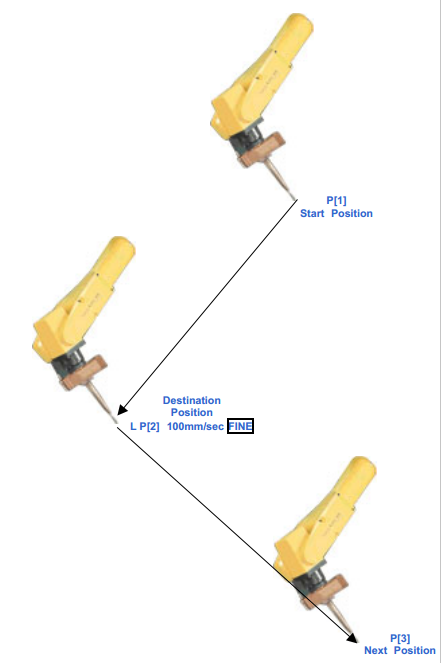
There are two types of motion termination, FINE and CNT (continuous).

To change between the two types, cursor to either of them and press F4 CHOICE.

**FINE termination**

With FINE termination selected, the robot visits each point exactly and momentarily stops

at each position before moving to the next position.



**CNT Termination**

With CNT termination selected, the robot curves each point, does not stop (at high values

of CNT) and may not visit the points exactly.

The continuous value can be between 0 and 100. This determines how close the robot

will come to the destination position.

At CNT100, the robot moves furthest from the destination position, with minimum

deceleration.



CNT100

**EDCMD Menu**

*Editing your programs, adding and deleting lines etc.*

The EDCMD menu is located above F5 and to view it you must be in a program.

If EDCMD is not above F5, press the NEXT key.

**The menu consists of:**

1.  Insert

2.  Delete

3.  Copy (and Paste)

4.  Find

5.  Replace

6.  Renumber

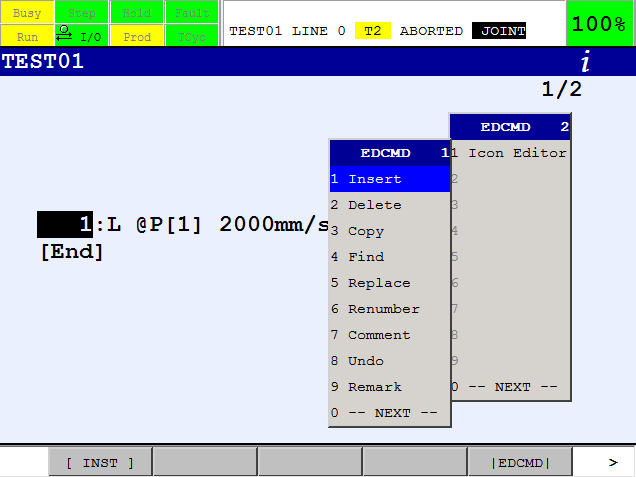
7.  Comment

8.  Undo

9. Remark

2-1. Icon Editor

This section will go through each item on the list and describe them in turn.



**Insert [EDCMD #1]**

If a further command or point needs to be added to an existing program, a blank line

needs to be inserted first, otherwise existing data will be lost.

**Using the Insert Function**

1. You are in the program screen

2. Place the cursor at the start of the program line below where you require the

inserted line to be.

***NOTE****: Lines will always be inserted above the cursor.*

3. If EDCMD is not above F5, press the NEXT key.

4. Press F5.

5. Select INSERT

6. Once Insert is selected a question will be asked:

How many lines to insert? :

7. Enter the number of lines you wish to insert, press ENTER and the blank lines will

appear.

**Delete [EDCMD #2]**

Deleting program lines.

**Using the Delete Function**

1. You are in the program screen
2. To delete a line or range of lines, place the cursor at the start of the program line at the top or bottom of the range.
3. If EDCMD is not above F5, press the NEXT key.
4. Press F5.
5. Select DELETE
6. The teach pendant will ask:

Delete line(s)   ? YES          NO

F4           F5

*Before selecting yes, highlight as many lines as you require to be deleted*

*by using the cursor keys ⇓ ⇑*

**Copy [EDCMD #3]**

Copying program lines.

**Using the Copy Function**

1. To copy, place the cursor at the start of the program line at the top or bottom of the

range of lines to be copied.

2. If EDCMD is not above F5, press the NEXT key.

3. Press F5.

4. Select COPY

5. COPY is above F2, and PASTE above F5.

6. Press F2 COPY

*Multiple lines can now be highlighted using the cursor keys.* ⇓⇑

7. When all the required lines are selected, press F2 COPY again. The copy has now

been performed.

**Paste**

Once program lines have been copied, they can be pasted as many times as required

and in as many different programs as required. You do not need to insert blank lines

before using the paste command.

1. Cursor to the start of the program line in the program where you wish to paste the

copied lines. It will paste ABOVE the cursor.

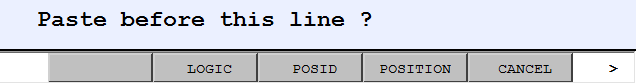
2. If PASTE is not above F5, press the NEXT key until EDCMD appears above F5.

Press F5, EDCMD and select copy.

3. Press F5 PASTE.

4. Once PASTE is selected, the following options will appear above the functions

keys.



***Leaves the position numbers blank. Does not record any positional information***

***Adds the lines exactly as they were and retains the current position numbers. Each pasted position will have the same identity as the original numbers***

***Adds the lines exactly as they were and renumbers the copied positions with the next available position numbers.***

*If copying from one program to another, you must select F4 POSITION to paste.*

*F3 POS-ID should only be used if* *you are pasting in the same program as you copied from.*

**Reverse Order Pasting**

To paste in reverse order press the NEXT key, after selecting the PASTE command. The

options above appear but this time with an R in front representing reverse.

***NOTE****: Copy and Paste will remain above the function keys until the prev key is pressed.*

**Find [EDCMD #4]**

In large programs the FIND command can be used to search the program for instructions

such as Registers, I/O and Labels among others.

The FIND command is located on the EDCMD menu.

**Using the Find Command**

1. You are in the program screen at the start of a program line

2. If EDCMD is not above F5, press the NEXT key.

3. Press F5.

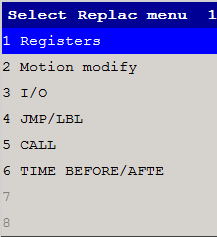
4. Select 4. FIND and follow the instructions on screen.

**Replace[EDCMD #5]**

The REPLACE command is used to replace instructions or motion

conditions/commands in a program.

It can be used to replace:



* FINE to CNT and vice versa
* Speed Values
* Speed Units
* Registers
* Jump Labels
* and others…

**Using the Replace Command**

1. You are in the program screen at the start of a program line
2. If EDCMD is not above F5, press the NEXT key.
3. Press F5.
4. Select 5. REPLACE and follow the instructions on screen.

**Renumber [EDCMD #6]**

The RENUMBER command is used to renumber positions sequentially. The positional

data will remain the same, only the position numbers will change.

Suppose you have a program containing 3 positions, like so:

J P[1] 100% FINE

J P[2] 100% FINE

J P[3] 100% FINE

You may wish to add a position between positions 1 and 2, (after inserting a blank line

first). When you do so, the new position will be given the next available position number,

like so:

J P[1] 100% FINE

J P[4] 100% FINE

J P[2] 100% FINE

J P[3] 100% FINE

The RENUMBER command allows you to renumber the positions sequentially like so:

J P[1] 100% FINE

J P[2] 100% FINE

J P[3] 100% FINE

J P[4] 100% FINE

The RENUMBER command is located on the EDCMD menu.

1. You are in the program screen at the start of any program line

2. If EDCMD is not above F5, press the NEXT key.

3. Press F5.

4. Select 6. RENUMBER

***NOTE****: The RENUMBER command changes the position number ONLY.*

**Comment [EDCMD #7]**

The COMMENT command enables or disables the appearance of comments on the

program screen.

For example, if a Register has a comment to describe it (SECTION 5), the comment will

appear in the program like so, if this command is enabled.

R[1: Counter] = R[1: Counter] +1

**Using the Comment Function**

1. You are in the program screen at the start of any program line

2. If EDCMD is not above F5, press the NEXT key.

3. Press F5.

4. Select 7. COMMENT

5. This will enable or disable the appearance of comments on the program screen.

**UNDO [EDCMD #8]**

The UNDO functions allows you to correct a mistake by undoing the last action.

***CAUTION****: An UNDO operation automatically rewrites the program, so that the results may not be those expected by the operator. Before executing a program after an undo operation, carefully check the program.*

*Notes*

*This function can undo the following operations*

*- Instruction modifications*

*- Line insertion*

*- Line deletion*

*- Copying of program statements*

*- Program instruction replacement*

*- Reassignment of position numbers*

*The UNDO function is disabled when any of the following operations are performed:*

*- Power Off*

*- Selection of another program*

**Using the UNDO Function**

1. You are in the program screen
2. If EDCMD is not above F5, press the NEXT key.
3. Press F5.
4. Select 8. UNDO

**Line Remark (Comment out) Instruction [EDCMD #9]**

The Line Remark instruction allows you to disable the execution and editing of an line (comment out).

Any instruction can be remarked. The LineRemark has the following characteristics:

* The Line Remark instruction will prefix the line with // and then display the line as normal.
* When the language is changed, the Line Remark instruction is localized.
* You cannot create a line with //. A line is created in the normal way.

You press [EDCMD] Remark to convert a line to a Line Remark.

* [EDCMD] Remark will allow multiple lines tobe selected and F4 REMARK or F5 UNREMARK

will remark or unremark all the selected lines.

* You cannot use // to comment out just a portion of a line.
* You cannot edit a line with //. Right arrow in the TP Editor will not move the cursor off the line number. TOUCHUP on a motion line is ignored.
* You can only remove the // using [EDCMD] Remark and selecting unremark.
* The line is still valid even though it is commented out. In other words, a position still exists for the motion lines and a label still exists for the label lines. In the case of a label, the label will be commented out so it is no longer valid to jump to. When a label is unremarked, then the label will be uncommented out unless another labelhas been added with the same number.
* You can perform other modifications on the line such as delete, copy and paste, find and replace, and renumber.
* Line Remark instructions must be valid instructions in order to successfully load a TP program or to perform Ascii Upload. You can use the Multi-lng Remark Instruction if you need to comment out an invalid instruction for Ascii Upload.
* The toolbars in the icon editor have a Line Remark Toggle button. When pressed, the current line is toggled as a remark/unremark.

**Icon Editor [EDCMD #2.1]**

If you have the optional touch screen available on your iPendant, you can edit a program using the icon editor. A USB mouse can also be used by connecting it to the USB port on the iPendant.

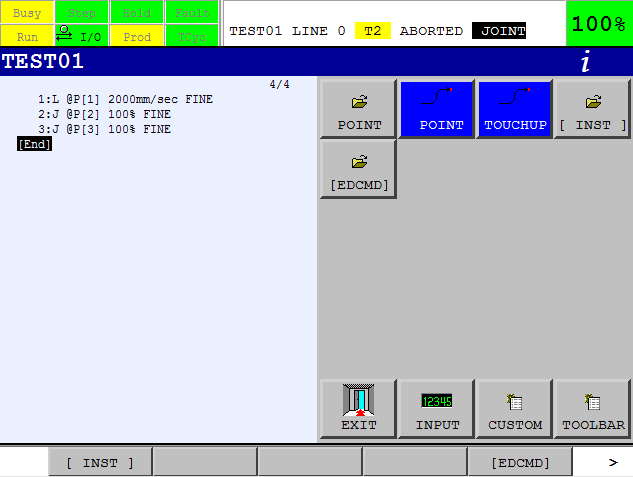
**Enter Icon Editor:**





1. Via the [EDCMD] menu, or by pressing + displays the Top Menu.
2. Press function key labelled PROGRAM [F2]
3. Select Icon Editor [#9] and the Icon Editor screen will appear.
4. You can also set the System Variable $UI\_CONFIG.$ICON\_EDIT[1] = TRUE.

The Editor will switch to an alternative mode which supports touch editing of icons.



When you select a different menu and return to the editor, the Icon Editor will be remembered.

To exit the Icon Editor and return to the Main Editor, press the EXIT icon.

If you do not have the optional touch screen available, you can also exit the Icon Editor by:





1. Press + , and the Top Menu is displayed.
2. Press function key PROGRAM [F2].
3. Select Main Editor [#8]
4. The Main Editor will now be used.

You can also set the System Variable $UI\_CONFIG.$ICON\_EDIT[1] = FALSE.

**INST Menu**

**INST>REGISTERS>**

**REGISTERS**

*Counting or storing a numerical value*

Registers may be thought of as value containers.

***NOTE****: A Register values is stored as a Double word (32 bits)*

They will store numbers given to them, via a program or manually, unil new values

are written.

There are 200 Data Registers as the default number on a R-30iB controller,

but this may be increased to 999 as desired (in Controlled start mode)

It is possible to specify registers as the direct and indirect addressing.

R [1] = 54

In a program they look like so 🡪

This statement means store a value of 54 in Register 1

**Using a Register as a Counter**

Registers can be used as counters to count production, or parts produced.

To use a register as a counter

R [1] = R[1] + 1

the following statement is required 🡪

Determines how many parts per cycle are being counted.

In this example 1 will be added into Register [1] every time the robot reads this

statement in a program.

Registers are found on the [INST] menu.

**Resetting a Register**

Registers store the last value given to them until told otherwise. Therefore, it may be

necessary to reset a register before using it. This will ensure that it begins its count at 0.

To reset any register, the program

R [9] = 0

line should look like this 🡪

This means set the value of register 9 to zero.

As an example, suppose we want to enter…

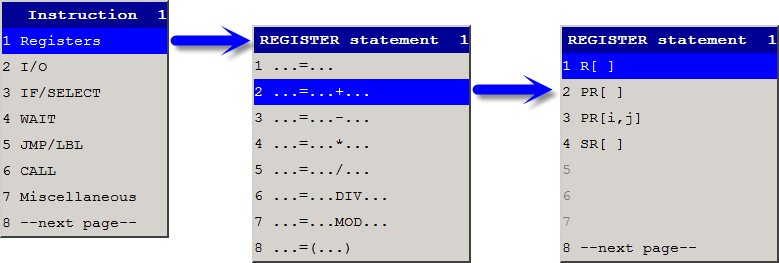
R [2] = R[2] + 1

…in to a program to set register 2 up as a counter.

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End]

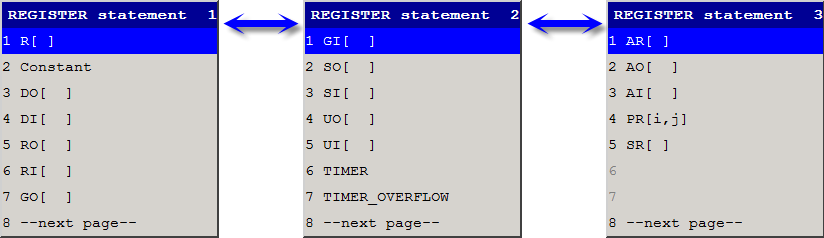
1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select item Registers
4. From the next screen select item 2  … = …+…
5. The cursor will now jump to the first position in the line and give a list of choices like that shown.



1. Select item 1. R [ ]
2. The cursor will jump between the two square brackets like so:

R [\_] = … + …

1. Enter the value of the register you wish to use.
2. The cursor will now jump to the other side of the = sign and give available choices:



1. Select item 1. R [ ]
2. The cursor will jump between the two square brackets. Enter the required register number.

R [2] = R[…] +….

1. Enter the same value of register as that before the = sign.
2. The program line should now look like so, with the cursor to the right of the + sign:

R [2] = R[2] +…

1. The REGISTER statement manu will appear, giving the available choices once again (as picture shown on previous page)
2. Now we need to choose 2. Constant, to enter a constant value of 1.

16. Enter 1 via the numeric keypad and the program line will now look like so:

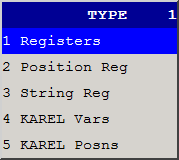
R [2] = R[2] + 1

**Viewing Registers**



1. Press the DATA key to view registers and position registers

NOTE: To return to the program screen from the data screen (or any other)press the EDIT key.



1. If registers are not displayed, press F1, TYPE

and select Registers. To change between Registers

and Position Registers, press F1 TYPE.

1. To change the value a register is storing, cursor to it,
2. Use the numeric keypad to insert a new number,

and press ENTER.

**Adding a Comment to a Register**

To add a comment to describe a register, cursor to the register and press ENTER. All the

letters of the alphabet should appear above the function keys. Type in the required

comment and press ENTER.

**INST>REGISTERS>POSITION REGISTERS**

**POSITION REGISTERS**

*Storing positional data away from a program*

***CAUTION****: Do not attempt to use postion registers until you have read all the available information and understood all the given instruction.*

A Position Register stores positional information, in the form of X, Y, Z, W, P and R.

Position Registers are a powerful programming tool because they can alter the way a

program runs. It is ESSENTIAL to understand Position Registers implicitly before you

begin programming with them using Offsets. Position Registers are extremely versatile

and have multiple uses.

***CAUTION****: Position Registers are affected by User Frames and Tool Frames.*

**Using Position Registers as Program Positions**

Position Registers are stored away from a program and as such can be used by multiple

programs.

An efficient programming technique is to store frequently used positions (such as the

Home position) in a Position Register.

This can then be called up from as many different programs as require it.

**Advantages**

You may at a later date wish to alter your Home Position.

If it is stored in a Position Register you need only alter the relevant position register and

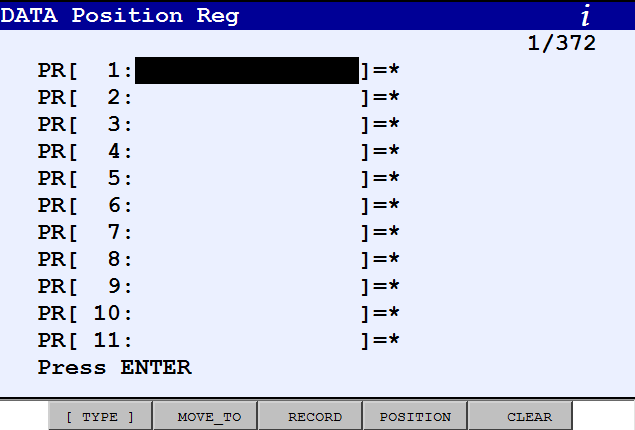
all programs using it will adjust accordingly.

However, if you have not stored your Home Position in a Position Register you will need

to go into every program using the Home Position and alter them individually.

**Recording Position Registers**

1. Press the DATA key
2. If Position Registers are not displayed,



press F1, [TYPE]

1. Select Position Reg.
2. This screen will be displayed 🡪
3. Cursor to the Position Register

you wish to record into.

1. Jog the Robot to the required position

and press SHIFT + F3, RECORD.

1. Use SHIFT + F2, MOVE\_TO to move to

the position of the current Position Register.

1. Use F5, CLEAR to clear the current Position Register.

**Adding to a Program**

To use the newly recorded position register in a program, record a position in space in

the usual way (SHIFT + F1, POINT) regardless of where the robot is.

1. Cursor onto the position number of the recorded position like so:

J P[ 1 ] 100% FINE

1. Press F4 CHOICE
2. From the options available in the menu that appears, select PR[ ]
3. Enter the Position Register number you record the position in.

**Adding a Comment to a Position Register**

You may wish to add a comment to a Position Register to describe it.

1. In the DATA screen, cursor to the Position Register you wish to add a comment to and press ENTER.
2. All the letters of the alphabet will appear above the function keys. Type in the required comment and press ENTER.

**INST>REGISTERS>POSITION REGISTERS**

**LPOS/JPOS**

*Recording the robots current position.*

The LPOS and JPOS commands allow you to store the current position of the robot into

a position register.

**Program Examples**

PR[5] = LPOS

LPOS stores the LINEAR (XYZWPR) coordinates of the robot

PR[5] = LPOS

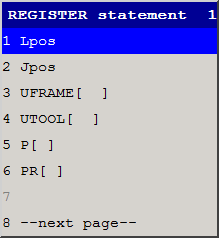
JPOS stores the JOINT (J1…J6) coordinates of the robot

**EXAMPLE INPUT**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select item Registers
4. From the next screen select item 1 … = …
5. The cursor will now jump to the first position in the line and give a list of choices .
6. Select item 2. PR [ ]
7. The cursor will jump between the two square brackets like so:

PR[. . .] = . . .



1. Enter the value of the position

register you wish to use.

1. The cursor will now jump to the other side of

the = sign and gives you a list of choises 🡪

**Inputs and Outputs**

**INST>I/O>**

**INPUTS AND OUTPUTS**

*Controlling and Monitoring Inputs and Outputs*

Output - Will turn something on or off. (e.g. a gripper)

Input -  Monitors signals. (e.g. a sensor)

There are various types of Inputs and Outputs (I/O):

**Digital**

* Digital I/O (DO/DI) signals provide access to data on a single input or output signal line.
* Signals can be ON or OFF.

**Robot**

* Robot digital input and output signals (RO/RI) are used to communicate between the controller and the robot.
* Signals are ON or OFF.

**Analogue**

* Analogue inputs and outputs (AI/AO) are continuous input and output signals and indicate values such as temperature and voltage.

**Group**

* Group inputs and outputs (GI/GO) are several digital input or output signals that have been assigned to a group.
* They are read as binary numbers.

**Group Input Example**

Suppose we have a number of digital inputs that we need to check and then have the

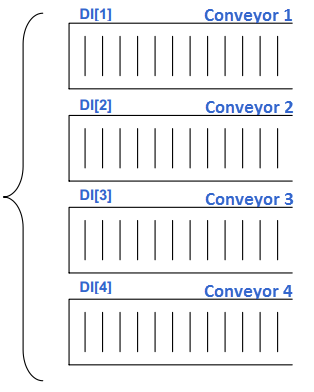
robot act accordingly.

For example, we have 4 conveyor belts, each with a digital input to check if a box is

waiting to be picked up by the robot.

Combine the 4 digital inputs into group input [1], GI[1].

DI[1] has group input value of 1



DI[2] has group input value of 2

DI[3] has group input value of 4

DI[4] has group input value of 8

Therefore:

DI[1] = ON ……………………………..  GI[1]= 1

DI[3] = ON ……………………………..  GI[1]= 4

DI[1] and DI[3] = ON ………………………. GI[1] = 5

DI[2], DI[3] and DI[4] = ON ……………. GI[1] = 14

This feature allows for efficient programming and

easy manipulation of digital inputs and outputs in a

program.

Group inputs and outputs can also

be assigned to registers.

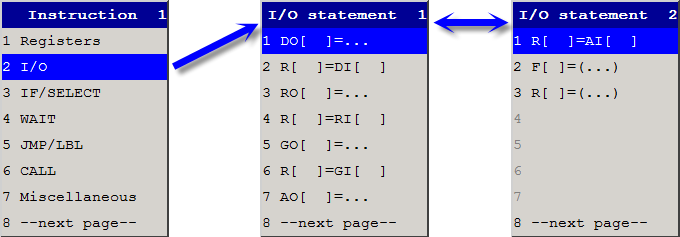
**Example input of I/O:**

Suppose we wanted to enter RO[3] = ON into our program to turn robot output number 3 on.

**Procedure**

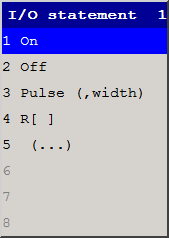
You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press INST (F1), and Select item 2. I/O, and this screen will be displayed



1. Select item 3. RO [ ] = ….
2. The cursor will jump between the two square brackets like so:

RO[. . .] = . . .



1. Enter the value of I/O you wish to use,

press ENTER and the cursor will jump

acrossto the right of the = sign and the

following choices will be displayed 🡪

1. Select item 1. ON
2. The program line will now look like so:

RO [3] = ON

1. This procedure may also be followed to turn off an output.

PULSE

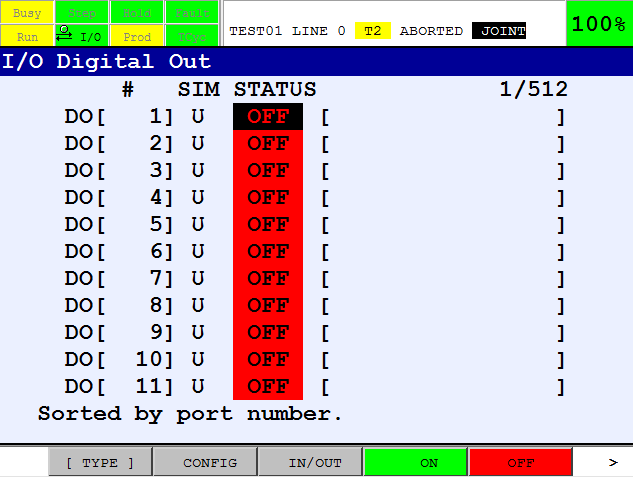
PULSE means the output will turn on for a specified time only.

For example, the line below Robot output number 3 would turn on for 2 seconds only.

RO [3] = PULSE 2 secs

**Viewing I/O’s**

1. Press MENUS
2. Select I/O
3. A screen similar to this should be displayed.



***To switch between the different types of output press F1 [TYPE], cursor to the correct output and press ENTER.***

***Outputs can be forced***

***On and OF by pressing***

***F4 and F5.***

**Adding Comments to I/O**

***Toggle between inputs and outputs screen by pressing F3***

For Robot I/O go to “Adding Comments to Robot IO” below

1. Press the MENUS key, and Select 5. I/O
2. If the I/O type you wish to add a comment to is not displayed,

press F1, [TYPE] and select the required type of I/O

1. Press NEXT until you see DETAIL at F4.
2. Press F4 DETAIL
3. At the Comment line, Press ENTER. Letters of the alphabet will appear above the function keys.
4. Using the function keys, enter a comment and then press ENTER

*To return to the program screen, press the EDIT key.*

Each time the input or output is called within a program, the comment should appear in

the program screen. If it does not you need to enable comments, See*the EDCMD Menu*.

**IF/SELECT Statement**

**INST>IF/SELECT>**

**IF STATEMENT**

*Setting up conditional statements*

The IF Statement allows you to carry out a program command, only when certain

conditions are met.

When the set conditions are met, the robot can call another program to run or

jump to a label.

As an example, we will set up an If statement to call another program when 3 cycles

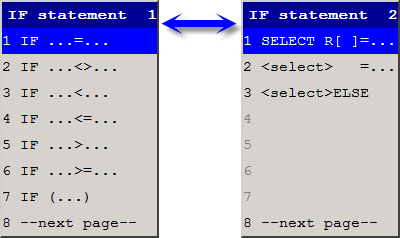
have been executed. Remember, a Register counter statement would be needed to

count the number of cycles.

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1,



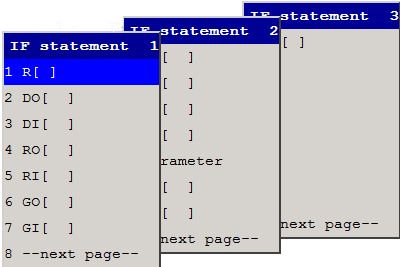
press the NEXT key.

1. Press F1 INST, and Select

item 3. IF/SELECT, and

this screen will be displayed 🡪

1. Select item 1. IF ... = …



1. The cursor will now jump to

the first position in the line

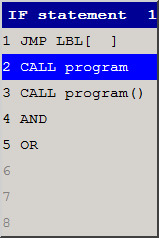
and give a list of choices 🡪

1. Select item 1. R [ ]
2. The cursor will jump between the two square brackets like so:

IF R[. . .] = … …

1. Enter the value of the register you wish to use and press ENTER.
2. The cursor will now jump to the other side of the = sign with a list of choises
3. Select item 2. Constant
4. Enter the value of the constant

you require and press ENTER.



1. The cursor will jump to the next space and

give you another menu to choose from. 🡪

1. Select item 2. Call program

and a list of all the teach pendant

programs will appear in alphabetical order.

1. The program line will now look like so:

IF R [1] = 3 CALL (name of program)

Although this example calls a program depending on the IF statement, a JMPLBL could also be used with the IF statement to jump to another part of the same program.

*REMEMBER: You must have the same register as that in the IF statement set up as a counter above the IF statement.*

**INST>IF/SELECT>**

**SELECT**

*Adding conditions into a program*

The SELECT statement is located on the INST menu and works in a similar but slightly

different way to the IF statement.

**How SELECT works:**

Imagine a program that needs to check a register value and then depending on the value

jump to different lines in the program.

Suppose we use Group Input 1, GI[1].

Using Group Input 1, GI[1], many IF statements would be necessary like so:

1. IF GI[1] = 2 CALL PROGRAM1

2. IF GI[1] = 4 CALL PROGRAM1

3. IF GI[1] = 8 CALL PROGRAM1

4. IF GI[1] = 16 JMP CALL PROGRAM1

4 different IF statements are necessary and the robot would read them all

until it found one that was true.

Instead, with the Select statement the program would work like so:

***The robot will branch***

***to whichever***

***statement is true.***

1. SELECT GI[1] = 2 CALL PROGRAM1

= 4 CALL PROGRAM2

= 8 CALL PROGRAM3

= 16 CALL PROGRAM4

**Example Input**

Entering a SELECT statement.

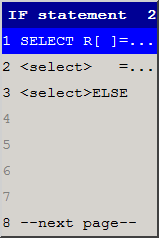
**Procedure**

1. You are in the program screen and have inserted a blank line or the cursor is on [End].
2. If INST is not above F1, press the NEXT key

***Choose item 1. SELECT R[ ]***

***when you are setting up a new SELECT statement.***

1. Press F1 INST and Select IF/SELECT



1. Select 8. next page 🡪

***Choose item 2 or 3 when you wish to add new branches to an existing SELECT statement***

*NOTE: To add a branch to an existing SELECT statement, the cursor must be placed on the line below the existing SELECT statement or below the last branch added to the SELECT statement.*

**WAIT INSTRUCTIONS**

**INST>WAIT>**

**WAIT INSTRUCTION**

*Pausing the program or waiting for a condition to become true*

The WAIT Instruction tells the robot to wait for something to happen.

The robot can wait for :

* A given time in seconds
* An input or output
* A register to be equal to a certain value.
* A semaphore

Wait is located on the INST menu.

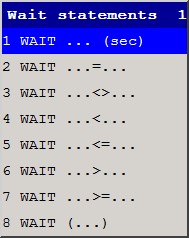
**Example Input**

As an example we will enter a statement to make the robot wait for 2 seconds before

carrying on to the next program line.

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].



1. If INST is not above F1, press the NEXT key.
2. Then press F1 INST, and select item 4. WAIT

and this screen will be displayed 🡪

1. Select item 1. WAIT … (sec)

And the program line will look like this:

WAIT 0.00 (sec)

1. Enter the number of seconds you wish to wait, and press ENTER.

The program line should look like this

WAIT 2.00 (sec)

*NOTE: The WAIT Instruction can also be used to wait for inputs to turn on, or to wait for a register to reach a certain value*

**LABELS & JUMP LABELS**

**INST>JMP/LBL>**

*Using labels and jump labels to create cycles*

Labels (LBL) are similar to markers in that they mark a point in a program for the robot to

jump to when the jump to label (JMPLBL) command is used. They are therefore used for

cycling purposes.

For example:

**1. LBL [1]**

2. J P[1] 100% FINE

3. J P[2] 100% FINE

4. J P[3] 100% FINE

5. J P[4] 100% FINE

**6. JMPLBL [1]**

***Every time the robot reads this line, it will jump to label [1], and hence run the entire program again and again.***

***This label defines the line for the robot to jump to when it reads a JMPLBL[1] command.***

The above is an example of an unconditional loop. It will continue to cycle no matter what. To set up a conditional loop, an IF statement is required in combination with a Register counter.

The order of the LBL and JMPLBL commands in a program does not matter. You can place a JMPLBL command anywhere in a program to jump to any other label, even if the label appears after the JMPLBL.

**Entering a LBL command. (LABEL)**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Selecting item 5. JMP/LBL gives you two choices:
4. JMP LBL [ ]

2. LBL [ ]

1. To enter a LBL, select 2. LBL [ ] and the value of the label may be entered via the numeric keypad.

There are 10,000 labels that can be used, but the same one cannot be used in the same program. You can jump to the label as many times as required, but there must be only one label with the same number in any one program. For example, only one LBL[99] or LBL[2], etc.

**Entering a JMP/LBL command. (JUMP LABEL)**

**Procedure**

1. To enter a JMP/LBL. Press F1, INST again, and select 5. JMP/LBL.
2. Select 1. JMP LBL [ ] and enter the value via the numeric keypad.

The above example sets up an unconditional cycle. i.e. it will cycle forever. To set up a conditional cycle, you need to use the IF statement together with registers, labels and jump labels.

**CALL INSTRUCTION**

**INST>CALL>**

*Calling programs from within programs*

In addition to using the CALL Statement in conjunction with the IF Statement, it can also

be used alone. It is located on the INST menu and in a program looks like so:

CALL (name of program)

**Entering a CALL command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Selecting item 6. CALL, gives you two choices:
   1. CALL program
   2. END
4. Select item 1. CALL program
5. There will then be a list of available teach pendant programs displayed,

select the program you wish to call.

*Remember: The robot always takes the shortest route from where it is to where it is going. Make sure the path between programs is a safe one to take.*

**Function - Abort All**

When a program is called and not fully executed, the robot remains in that program.

You cannot get back to the host



program until you pressthe FCTN key and select 1. ABORT ALL.

*REMEMBER: the robot always takes the shortest route from where it is to where it is going. If you abort a program remember to jog the robot to a safe position away from a workpiece so that the route taken will be a safe one when you restart a program.*

**MISCELLANEOUS MENU**

**INST>MISCELLANEOUS>**

**RSR INSTRUCTION**

*Enables or Disables the RSR function.*

A Robot Service Request(RSR) is a request for service from an external device.

That request comes from a digital input signal on a preassigned RSR input line.

You can use up to eight robot servicerequest signals:

RSR1, RSR2, RSR3, RSR4, RSR5, RSR6, RSR7, and RSR8.

When the robot controller receives a service requestsignal, the controller determines whether the signal is acceptable. If acceptable, the controller determines which program to execute.

**INST>MISCELLANEOUS>**

**USER ALARMS**

*Setting up alarms to alert the user*

The Setting User Alarm screen allows you to define a message that will be displayed on the teach pendant status line. This message is displayed when a user alarm instruction is executed in a teach pendant program.

For example, if you define the message of user alarm1 (UALM[1]) to be "Perform repair procedure" and the instruction UALM[1] is executed in a teach pendant program, then the message will be displayed on the status line of the teach pendant as:

INTP-213 Perform repair procedure (name, line) UALM[1]

Where *name* is the name of the current program and *line* is the line number in which the UALM[1] instruction was executed.

**User Alarm Severity**

By default, the severity of a user alarm is STOP, which pauses the program and stops robot motion. If you want to change the severity of the user alarm, you must set the appropriate $UALRM\_SEV[n] system variable to a value that corresponds to the severity you want. "n" corresponds to the number of the user alarm

$UALRM\_SEV[n] Severity Values:

|  |  |
| --- | --- |
| **Value** | **Action** |
| 0 | No action |
| 2 | Pause program |
| 3 | Abort program with error |
| 4 | Stop program motion |
| 6 | Pause program and stop its motion |
| 8 | Cancel program motion |
| 10 | Pause program and cancel its motion |
| 11 | Abort program and cancel its motion |

16 added to any value causes servomotors to be turned off.

32 added to any value causes the action to apply to all programs and all motions.

64 added to any value requires a Cold start to reset the controller.

*The default value of all the user alarms is 6.*

For example,

* A value of 0 causes a warning message to be displayed.
* A value of 6 pauses the program and stops its motion.
* A value of 43 aborts all programs and cancels all motions (11 + 32

**Changing the severity of a user alarm.**

To change the severity of a user alarm, the value stored in the system variable

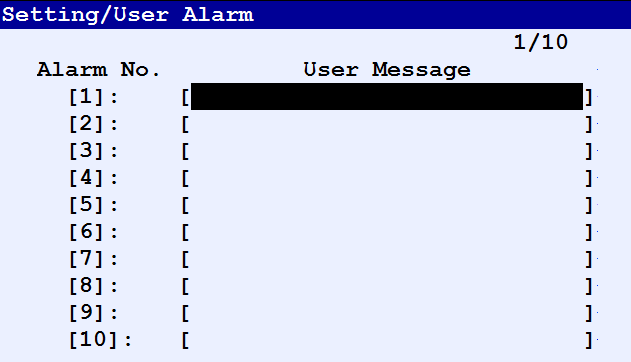
$UALRM\_SEV must be changed.

**Procedure**

1. Press the MENUS key and 0. next page
2. Select 6. System.
3. Select 2. Variables
4. Cursor to $UALRM\_SEV and press ENTER
5. The severities of all 10 user alarms is shown.
6. Cursor to the user alarm you wish to change and enter a new value with the numeric keypad.

**Entering the User Alarm Messages**

1. Press the MENUS key.
2. Select 6. System
3. Select 8.User Alarm - and the following screen will be displayed showing the 10 user alarms.



1. Cursor to the alarm you wish to use and press ENTER.
2. The letters of the alphabet will appear above the function keys. Enter the message you require and press ENTER.
3. The message is now set and when a program encounters a UALM instruction calling that user alarm, the message will appear at the top of the teach pendant informing the user of the fault.

**Entering a User Alarm in a program.**

**Procedure**

1. Press the MENUS key and 0. next page Select 6. System
2. You are in the program screen and have inserted a blank line or the cursor is on [End].
3. If INST is not above F1, press the NEXT key.
4. Press F1, INST
5. You need to select Miscellaneous from the INST menu.

(You may need to press 8. next page first.)

1. From the Miscellaneous menu, select UALM[ ]
2. It will appear in your program like so:

UALM […]

1. Enter the value of alarm you wish to use and press ENTER.

**INST>MISCELLANEOUS>**

**TIMERS**

*Timing all or part of a program*

Timers can be used to time all or any part of a program.

A timer will store the last time recorded into it, even after the controller has been

switched off.

When it is next used, it will continue to add onto that time and so will not give a

true reading of how long a program took to execute.

It is for this reason that a timer must be reset before it is used.

Timers are located on the Instruction Menu [INST] and in a program will look like so:

**1. TIMER [2] = RESET**

**2. TIMER [2] = START**

3. J P[1] 100% FINE

4. J P[2] 100% FINE

5. L P[3] 2000mm/sec FINE

6. L P[4] 2000mm/sec FINE

**7. TIMER [2] = STOP**

**Entering a TIMER command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Miscellaneous from the INST menu.

(You may need to press 8. next page first.)

1. Once in the Miscellaneous menu, select TIMER[ ]
2. It will appear in your program like so:

TIMER [ ] = . . .

1. Enter the value of timer you wish to use and press ENTER.
2. The cursor will jump over the = sign and give you the following choices:

1. RESET

2. START

3. STOP

1. Choose the item you require.

*Remember, you must reset a timer before starting it.*

**Viewing Timers**

1. To view the timer recorded by a timer in a program, press the MENUS key.
2. Select 0. next page
3. Select Status
4. Press F1 TYPE and select Prg Timer.

**INST>MISCELLANEOUS>**

**OVERRIDE**

*Control the override speed with a program instruction.*

The override instruction allows you to control, through a program instruction, the override

speed the robot runs at. Values available are 0 – 100%.

**Example**

J P[1] 100% FINE

L P[2] 2000mm/sec FINE

L P[3] 2000mm/sec FINE

**OVERRIDE 100%**

L P[4] 2000mm/sec FINE

L P[5] 2000mm/sec FINE

L P[6] 2000mm/sec FINE

L P[7] 2000mm/sec FINE

**OVERRIDE 20%**

L P[8] 2000mm/sec FINE

L P[9] 2000mm/sec FINE

L P[10] 2000mm/sec FINE

[End]

**Entering an OVERRIDE command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Miscellaneous from the INST menu. (You may need to press 8. next page first.)
4. Once in the Miscellaneous menu, select OVERRIDE[ ]
5. Choose input options: 1. R[ ] 2. Constant 3. AR
6. Chose e.g. Constant, and It will appear in your program like so:

OVERRIDE = (value 0-100)

1. Enter the value you require and press ENTER.

**INST>MISCELLANEOUS>**

**REMARK**

*Adding a comment into a program*

Remarks are basically comments that you can add at any point to a program. A remark

will make no difference to a program because the robot ignores them. They merely serve

as a way for the programmer to add a comment that may later aid memory as to what

each line of the program does.

Lines of code in a teach pendant program do not make for easy reading to anybody

unfamiliar to it. Adding comments at appropriate points to explain what a program is

doing will serve as a guide to what the program does to a user.

MPLE INPUT

**Adding a Remark to a Program.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key
2. Press F1 INST
3. Select Miscellaneous from the INST menu. (You may need to press 8. next page first.)
4. Once in the Miscellaneous menu, select Remark
5. It will appear in your program like so:

! \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Press Enter and the letters of the alphabet will appear above the function keys and allow you to enter a comment.

***Remembe****r, the robot does not read the comment during program execution.*

**INST>MISCELLANEOUS>**

**MESSAGE**

*Allows you to display a message on the user screen.*

The message instruction displays a specified message on the user screen.

**User Screen**

The user screen displays messages from a program being executed.

To view the User screen:

1. Press the MENUS key.

2. Select User.

***Note****: When a message instruction is not executed, nothing appears on this screen.*

A message can consist of up to 24 characters including alphanumeric characters,

asterisks (\*), underlines(\_) and ampersand marks, (@).

**Entering a MESSAGE command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Miscellaneous from the INST menu. (You may need to press 8. next page first.)
4. Once in the Miscellaneous menu, select MESSAGE[ ]
5. It will appear in your program like so:

MESSAGE [\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ]

Press ENTER and user the function keys to enter a message.

**INST>MISCELLANEOUS>**

**PARAMETER**

**INSTRUCTION**

*Changing the value of system variables.*

The parameter instruction allows the value of a system variable to be changed. This

instruction can be used for a system variable containing a numeric value only.

***NOTE:*** *The operation of the robot and control unit is controlled with system variables. Only a person who knows details of the influence of changes in system variables should set system variables. If a person without detailed knowledge attempts to set the system variables, the robot and control unit could malfunction, causing injury.*

**System Variable Notation**

System Variables are represented by a dollar sign [$] at the beginning of them

**System Variable Types**

There are 2 types of system variables:

**Variable type**

These can be assigned to a register.

**Position type**

These can be assigned to a position register. There are 3 data types for a position type system variable:

* Orthogonal [XYZWPR]
* Joint[J1, J2, J3, J4, J5, J6]
* Matrix[AONL]

When a system variable of position type is assigned to a position register, the position register is automatically converted to the correct data type of the system variable.

If system variables are incorrectly assigned, i.e if a variable type is assigned to a position register or a position type is assigned to a register, the following alarm will be displayed.

INTP – 240 Incomputible datatype

**Entering a parameter instruction.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Miscellaneous from the INST menu. (You may need to press 8. next page first.)
4. In the Miscellaneous menu, select 7. Parameter name
5. It will appear in your program like so:

$ … = …

1. Press ENTER and user the function keys to enter the name of the system variable.

You can either write to or read from a system variable.

**Writing to System Variable**

$ASCII\_SAVE=1

This will set the system variable $ASCII\_SAVE to a value of 1

**Reading From a System Variable**

R[1] = $ASCII\_SAVE

This will put the value of $ASCII\_SAVE into Register [1]

**INST>MISCELLANEOUS>**

**MAXIMUM SPEED INSTRUCTION**

*Changing the value of system variables.*

The maximum speed instruction allows you to specify the maximum operating speed of a

program. There are two maximum speed instructions:

JOINT\_MAX\_SPEED

Allows you specify the joint operation speed.

LINEAR\_MAX\_SPEED

Allow you to specify the path control operating speed.

If a user selects a speed exceeding the value set in the program it will be ignored and the

speed set with the maximum speed instruction is used.

EXAMPLE INPUT

**Entering a maximum speed instruction.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Miscellaneous from the INST menu. (You may need to press 8. next page first.)
4. In the Miscellaneous menu, select Max Speed Instruction
5. Select either JOINT\_MAX\_SPEED or LINEAR\_MAX\_SPEED
6. Examples:

JOINT\_MAX\_SPEED[1] = 10

LINEAR\_MAX\_SPEED = 100

**Multiple Control Menu**

**INST>MULTIPLE CONTROL>**

**RUN COMMAND**

*Running two programs at once.*

The Run command allows to execute a program from within another program so that

both programs run together.

You cannot run 2 programs together that use the same motion group, that is the same

axes. The programs must use different motion groups, or the second program is set up

as a no motion program. That is it doesn’t have any positional commands in it.

To synchronise programs, the semaphore instruction is used.

**Differences Between CALL and RUN**

The CALL instruction causes the robot to branch out of the program it is currently in and

run the program being called.

The RUN instruction remains in the program it is in but also runs the program that is

called at the same time. The two programs run concurrently.

**Program**

**PROGRAM1**

1. J P[1] 100% FINE

2. RUN PROGRAM2

3. J P[2] 100% FINE

4. J P[3] 100% FINE

**MOTION GROUP [1, \*, \*, \*, \*]**

**PROGRAM2**

1. DO[1]=ON

2. RO[5]=ON

3. RO[6]=ON

4. RO[7]=ON

**MOTION GROUP [\*, \*, \*, \*, \*]**

**Example**

**Entering a RUN command.**

**Procedure**

1. Go to the INST menu, and select Multiple Control,
2. select Run Instruction

RUN (name of program)

1. It will appear in your program like so:
2. A list of all the available teach pendant programs will appear.

Select the program you wish to run.

**INST>MULTIPLE CONTROL>**

**SEMAPHORE**

**INSTRUCTION**

*Used when programs are multi-tasking.*

The Semaphore instruction switches a semaphore, specified with a number, between on

and off. A semaphore is a kind of switch used to synchronise the execution of tasks.

**Program Example**

PROGRAM2

1. J P[1] 100% FINE

2. J P[2] 100% FINE

3. J P[3] 100% FINE

4. J P[4] 100% FINE

5. SEMAPHORE[1]=ON

This program will turn SEMAPHORE[1] on.

**PROGRAM1**

1. SEMAPHORE[1]=OFF

2. RUN PROGRAM2

3. J P[1] 100% FINE

4. J P[2] 100% FINE

5. WAIT SEMAPHORE[1]

This program will wait at line 5

for SEMAPHORE[1] to be ON

The SEMAPHORE command can be used with a TIMEOUT like so:

WAIT SEMAPHORE[1] TIMEOUT, LBL[1]

**Entering a SEMAPHORE command.**

**Procedure**

1. You are in the program screen and have inserted a blank line or the cursor is on [End].
2. If INST is not above F1, press the NEXT key.
3. Press F1 INST
4. Select Multiple Control from the INST menu. (You may need to press 8. next page first.)
5. In the Multiple Control menu, select Semaphore or Semaphore Wait Instruction

**Program Control Instruction**

**INST>PROGRAM CONTROL>**

**HALT**

**INSTRUCTION**

*Interrupts program execution.*

The halt instruction allows you to stop robot motion in the following way:

* If an operation instruction is being executed, the program stops before the operation is completed.
* The cursor moves to the next line. When restarted the program is executed from this line.
* If the program timer is active, it is stopped. When the program is restarted, the program timer is activated.
* If a pulse output instruction is being executed, the program stops after that instruction has been executed.
* If an instruction other than a program call instruction is being executed, the program stops after that instruction has been executed. A program call instruction is executed when the program is restarted.

Halt Instruction:

PAUSE

**Entering a PAUSE command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Program Control from the INST menu. (You may need to press 8. next page first.)
4. In the Program Control menu, select PAUSE

**INST>PROGRAM CONTROL>**

**ABORT**

**INSTRUCTION**

*Aborts program execution.*

The abort instruction aborts program execution in the following way:

* If an operation instruction is being executed, the program stops before the operation is completed.
* The cursor stops on the current line. When restarted the program is executed from this line.
* When the abort instruction is executed, the execution of the program CANNOT be continued.
* Information held by a program call instruction about the main program is lost.

Abort Instruction

ABORT

**Entering an ABORT command.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Program Control from the INST menu. (You may need to press 8. next page first.)
4. In the Program Control menu, select ABORT

**OFFSETS MENU**

**INST>OFFSETS**

*Mathematically altering a program*

Offsets are a way of either offsetting a single position, multiple positions or a whole

program without the need for reprogramming. Offsets are a useful way of incrementing

program positions by direct measurements.

For example, palletising requires that a program increment vertically to form the layers.

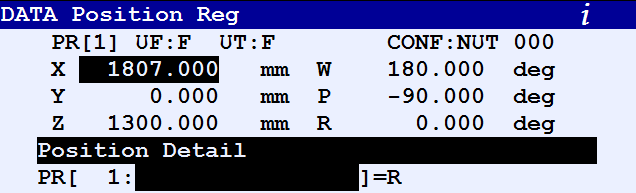
To use offsets, you need to understand Position Register Elements.

**Position Register Elements**

Understanding Position Register Elements is the key to a powerful programming tool.

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select Position Reg
4. Press F4, POSITION and this screen will be displayed.

Showing the six elements of a position register:



***Element 1***

***Element 4***

***Element 2***

***Element 5***

***Element 3***

***Element 6***

The 6 elements of a Position Register are X, Y, Z, W, P and R

They each store a piece of information that together identifies the position register.

Each element can be addressed either from the DATA screen, or from a program.

**Addressing Position Register Elements from a Program**

The program command for addressing individual elements looks likes so:

PR [ i , j ]

***i is the Number of the Position Register you wish to address***

***j is the Position Register Element you wish to address***

**Example Program Command**

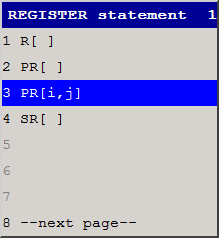
***Means: send a value of 200mm into the Z element of Position Register [5].***

PR [ 5, 3 ] = 200

If the Position Register is represented as a Joint move instead of Cartesian, this command would send in 200 degrees to Joint 3.

**Entering This Command into a Program**

This command is available on the instruction menu like so:



1. You are in the program screen
2. Press F1, INST
3. Select Registers
4. Select item 1. …=…
5. Select PR[i,j] from the menu that appears 🡪

**Adding an Offset to a Program Line**

1. Cursor to the end of the position you wish to use the offset with:

J P[1] 100% FINE \_\_\_\_

1. Press F4 CHOICE, and select e.g. Offset, PR [ ]

J P[1] 100% FINE Offset, PR [5]

This program line means add on the ENTIRE contents of Position Register [5] to programmed position [1] and move there.

ALL of the information in PR[5] is used for the offset. It is therefore important to ensure that the other elements of PR[5] are set to zero if you do not wish to offset by them

**INST>OFFSET>**

**FRAMES**

**INSTRUCTION**

*Change the frame setting*

The FRAME instruction is used to change the setting of the Cartesian coordinate system

by which the robot works.

There are two kinds of FRAME instruction.

**Making a Frame Equal to a Position Register**

The definition of the specified frame is changed.

***NOTE:*** *Please ensure you properly understand the implications of making a frame equal to a position register*

**Tool Frame**

The tool frame setup instruction changes the setting of the tool frame you

select and makes it equal to a position register.

UTOOL […] = PR[...]

***Put the Tool Frame number here you wish to use***

***Put the position register you wish to make it equal to here.***

**User Frame**

The user frame setup instruction changes the setting of the user frame you

UFRAME […] = PR[...]

select and makes it equal to a position register.

***Put the position register you wish to make it equal to here.***

***Put the User Frame number here you wish to use***

**Activating a Frame from within a Program**

**Tool Frame**

This changes the tool frame number being selected.

***You can make the Tool Frame equal to a constant or a register.***

***This means make Tool Frame 1 the Active Frame. This would usually appear at the very top of a program.***

UTOOL NUM = 1

Program Example:

**User Frame**

This changes the User frame number being selected.

***You can make the User Frame equal to a constant or a register.***

UFRAME NUM = 1

Program Example:

***This means make User Frame 1 the Active Frame. This would usually appear at the very top of a program.***

***NOTE:*** *If you activate a tool or user frame that is different to the one in which a program was created, you will generate an error and the robot will not move.*

***WARNING:*** *If you change the values in a user or tool frame and then try and run a program that was created with the previous user and tool frame values, the robot will move relative to the new frame information. ENSURE YOU UNDERSTAND THE IMPLICATIONS OF THIS. THE ROBOT MAY NOT MOVE WHERE YOU EXPECTED IT TO.*

**Entering a Frame Instruction.**

**Procedure**

1. You are in the program screen and have inserted a blank line or the cursor is on [End].
2. If INST is not above F1, press the NEXT key.
3. Press F1 INST
4. Select Offset from the INST menu. (You may need to press 8. next page first.)
5. In the Offset menu, select the frame instruction you require.

**SKIP CONDITION**

**INST>SKIP>**

*Specifying in advance a skip instruction for the robot to follow.*

A skip*condition* is used with a skip*instruction*.

Before a skip instruction can be executed, a skip condition instruction must be executed.

A skip condition once specified is valid until the execution of the program is completed, or

the next skip condition instruction is executed.

***SKIP CONDITION DI[5] = ON***

***Note: In the skip condition, multiple conditions can be specified on a single line use ‘AND’ and ‘OR’***

**PROGRAM Example**

1. SKIP CONDITION DI[5] = ON

2. J P[1] 100% FINE

3. L P[2] 2000mm/sec FINE, Skip, LBL[1]

4. J P[3] 100% FINE

5. LBL[1]

6. J P[4] 100% FINE

***SKIP INSTRUCTION: If the Skip Condition is NOT met, the robot at this point will jump to label [1]. If the Skip Condition IS met, the robot at this point will continue to the next line.***

**Entering a Skip Condition.**

**Procedure**

You are in the program screen and have inserted a blank line or the cursor is on [End].

1. If INST is not above F1, press the NEXT key.
2. Press F1 INST
3. Select SKIP from the INST menu. (You may need to press 8. next page first.)
4. Select SKIP CONDITION

**Entering a Skip Instruction.**

**Procedure**

1. Cursor to the end of the program line you wish to add a skip instruction to like so:
2. Press F4 CHOICE

J P[1] 100% FINE \_\_\_

1. Select Skip, LBL[ ]
2. Enter the label number you wish to use.

**FRAMES OVERVIEW**

*General overview of frames, see individual frame sections for further detail*

Frames are used to describe the location and orientation of a position.

The LOCATION is the distance x, y, and z directions from the origin of the reference

frame. The location is expressed in millimetres.

**NOTE:** YOU MUST USE THE SAME REFERENCE FRAME THAT YOU RECORDED A POSITION IN WHEN YOU EXECUTE A PROGRAM.

The ORIENTATION is the rotation about the x, y, and z axes of the reference frame,

called w, p, and r. The orientation is expressed in degrees.

When a position is recorded, the location and orientation of the robot are automatically

recorded, relative to the active reference frame.

**NEVER OVERWRITE** a reference frame that is being used by a program. Damage to equipment can occur or injury to personnel.

The robot system has four kinds of frame, World, User, Jgfrm and Tool.

World is a default of the robot and cannot be changed

**Three Types of Frame**

There are three different types of frames that can be set up in the robot:

* Tool Frames
* User Frames
* Jog Frames

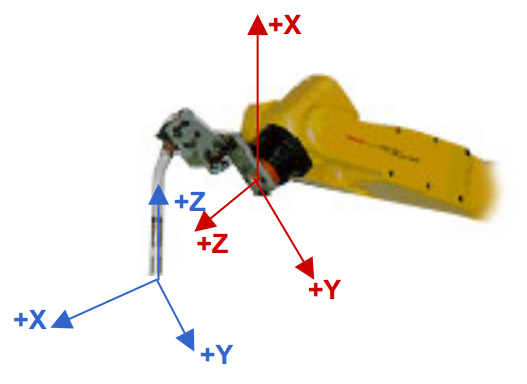
**IMPORTANT:** Frames can alter the way the robot moves to positions. Please ensure that you fully understand how to set up frames and the implication changing frames can have on program positions.

**TOOL FRAMES**

*Setting up the Tool Center Point (TCP)*

By default, the origin of the tool frame is on the faceplate of the robot.

You must move the origin of the tool



frame to the position, both location and

orientation, where the work is to be

done.

This position is called the

Tool Center point (TCP).

All measurements in tool frame

are relative to the origin of

the tool frame (the TCP).

**Three methods for Setting up the Tool Center Point**

**Three Point Method**

* Use this to define the location of the tool frame.
* 3 Approach points are taught with the tool touching a common point from three different directions.

**Six Point Method**

* Use this to define the location and orientation of the tool frame.
* This six point method involves the same 3 approach points of the Three Point Method, but in addition 3 directional points are required.

**Direct Entry Method**

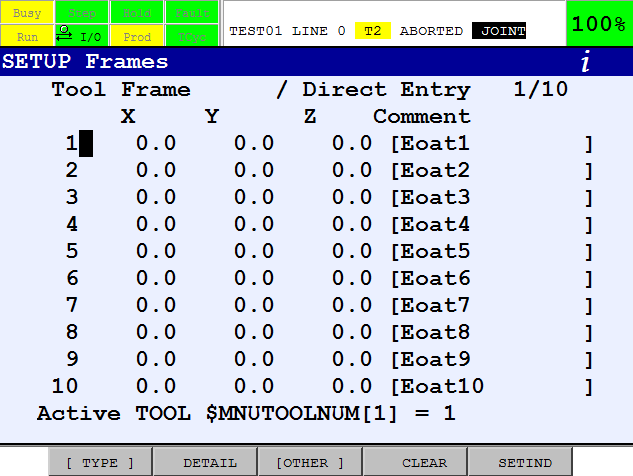
* This method provides direct recording and numerical entry of the frame position. The manufacturing specifications of the tool are required for this method.

**The Three Point Method**

***CAUTION****: Do not accidentally alter an existing tool frame or Tool Center Point.*

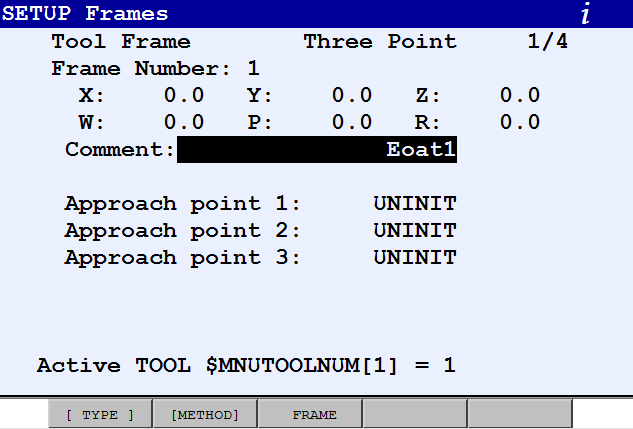
*- Doing so could lead to serious damage to equipment or injury to personnel.*

1. Press MENUS
2. Select SETUP
3. F1 [TYPE]
4. Select Frames
5. If tool frames are not displayed, press F3 [other], and select tool frames, this screen should appear.



1. Cursor to the frame to be set up and if necessary clear it by selecting F4 CLEAR
2. Select F2 DETAIL.
3. Press F2 METHOD
4. Select Three Point

The teach pendant screen should now be like this…



**Approach Point 1**

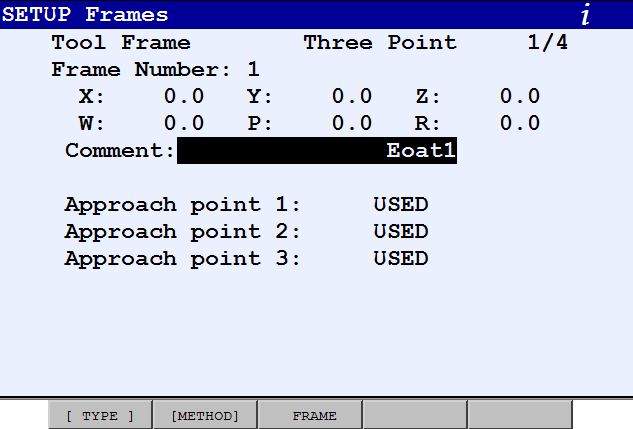
1. Cursor to Approach Point 1
2. Jog the robot so that the end of the tool touches the tip of the common point.
3. Press SHIFT + RECORD

**Approach Point 2**

1. Cursor to Approach Point 2
2. In JOINT, rotate axis 6 by more than 90° but less than 360°
3. In WORLD,USER or JGFRM, use the major robot axes to re-align the tip to the common point.
4. Press SHIFT + RECORD

**Approach Point 3**

1. Cursor to Approach Point 3
2. In JOINT, rotate axis 4 or 5 by less than 90°
3. In WORLD, USER or JGFRM, use the major robot axes to re-align the tip to the common point.
4. Press SHIFT + RECORD
5. The screen will now be like this:



***NOTE:*** *When the 3 point TCP is set up,*

***+Y becomes –Y***

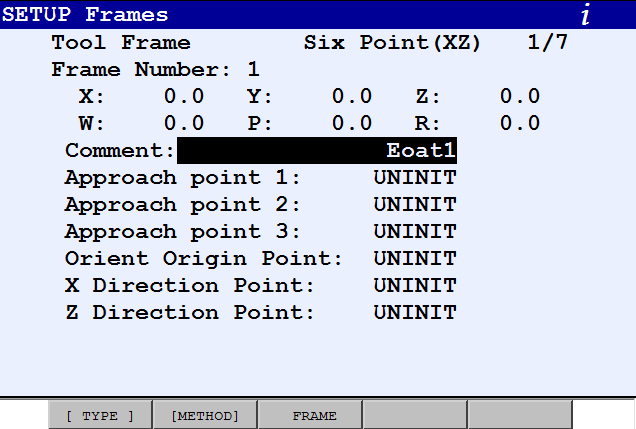
***+Z becomes –Z***

**The Six Point Method**

***CAUTION****: Do not accidentally alter an existing tool frame or Tool Center Point.*

*- Doing so could lead to serious damage to equipment or injury to personnel.*

1. Press MENUS, and Select 6.SETUP
2. Press F1 [TYPE], and Select 3.Frames
3. If tool frames are not displayed, press F3 [other], and select tool frames.



1. Cursor to the frame to be set up and if

necessary clear it by selecting F4 CLEAR

1. Select F2 DETAIL.
2. Press F2 METHOD
3. Select Six Point (XZ)

and the entry screen will be displayed 🡪

***NOTE:*** *You can convert a 3 point tool center point that has been set up to a 6 point tool center point. Go into the DETAIL of the 3 point TCP that has been set up and press F2 METHOD and select 6 point. The first 3 points will already be recorded and the last 3 points are available to record. Follow this procedure from ORIENT ORIGIN POINT*

**Approach Point 1**

1. Cursor to Approach Point 1
2. Jog the robot so that the end of the tool touches the tip of the common point.
3. Press SHIFT + RECORD

**Approach Point 2**

1. Cursor to Approach Point 2
2. In JOINT, rotate axis 6 by more than 90° but less than 180°
3. In WORLD,USER or JGFRM, use the major robot axes to re-align the tip to the common point.
4. Press SHIFT + RECORD

**Approach Point 3**

1. Cursor to Approach Point 3
2. In JOINT, rotate axis 4 or 5 by less than 90°
3. In WORLD, USER or JGFRM, use the major robot axes to re-align the tip to the common point.
4. Press SHIFT + RECORD

**Orient Origin Point**

1. Cursor to Orient Origin Point
2. In JOINT, jog the robot so that the tool is parallel to the Z axis of the robot WORLD frame, that is so the tool is vertical.
3. Ensure the tool is lined up with the X-axis.
4. Press SHIFT + RECORD

**X Direction Point**

1. Cursor to X Direction Point
2. In WORLD or JGFRM jog the robot in +X direction by at least 250mm.
3. Press SHIFT and RECORD.

**Z Direction Point**

1. Cursor to Orient Origin Point
2. Press SHIFT and MOVE\_TO(F4)
3. Cursor to Z Direction Point
4. In WORLD or JGFRM jog the robot in +Z direction by at least 250mm.
5. Press SHIFT and RECORD.



The Tool Center Point is now set up 🡪

The minor axes should make the robot

pivot around the TCP. The major axes will

move relative to the orientation of the tool.

The direction of the major axes will

depend on how they were set up.

**Making a Tool Center Point Active**

A Tool Center Point must be made active before it can be used. There are three ways of achieving an active TCP. It can be done manually by entering the setup screen, it can be set active via a program or there is a shortcut to setting it active via the COORD key. The three methods are detailed below.

You must run a program with the TCP it was created with active. If you try to run a program created in TCP 1 say, with TCP 2 active, you will generate an error message telling you that an invalid UTOOL is active.

**Manually setting a TCP active**

***NOTE****: A program created in one Tool Center Point will not run with another Tool Center Point active. Ensure the correct Tool Center Point is active BEFORE creating a new program.*

1. Press the PREV key and select [F5] SETIND
2. Enter the frame number you wish to make active and press ENTER.
3. Select the Tool Coordinate system using the COORD key, and the robot will now jog relative to the Tool Center Point. Jogging with the minor axes in tool frame

will make the tool pivot about the Tool Center Point.

- All the major axes will now move relative to the orientation of the tool.

**Setting a TCP active within a Program**

1. A Tool Center Point can be made active via the program screen.
2. To do this press INST
3. Select Offset/Frames (you may need to press 8. --next page-- first)
4. Select UTOOL\_NUM = ...
5. Select Constant from the choices given
6. Finally enter the value of the Tool Center Point required.

***NOTE****: A good programming technique for ensuring the correct Tool Center Point is active when a new program is being created is to complete the following program line at the very start of a program, before any points have been recorded.*

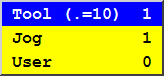
*Run the line using SHIFT + FWD, and this will ensure that the correct Tool Center Point is active before any program points are recorded.*

UTOOL\_NUM = (tcp number)

You must run a program with the TCP it was created with active. If you try to run a program created in TCP 1 say, with TCP 2 active, you will generate an error message telling you that an invalid UTOOL is active.

**Shortcut to Making a TCP Active**

1. Press and hold the SHIFT key and Press COORD
2. A small box will appear in



the top right hand corner of

the teach pendant screen 🡪

1. Cursor to Tool and enter

the frame number you wish to use with the numeric keypad.

**USER FRAMES**

*Setting up a user frame*

A user frame is a frame that can be created to allow for easy jogging in a plane that is different from the WORLD Frame.

When a User frame is created (UNLIKE a jog frame) a new zero point is defined.

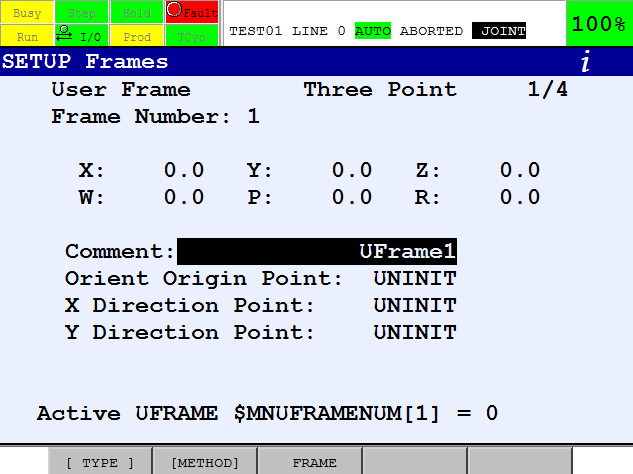
***NOTE****: Position registers and positions are recorded relative to the user frame. It is therefore essential to understand how user frames work and how a user frames can change the way a program runs.*

**The Three Point Method**

1. Press MENUS
2. Select SETUP
3. F1 [TYPE], select Frames
4. If user frames are not displayed, press F3 [OTHER], and select user frames
5. If necessary clear the frame by pressing F4 CLEAR

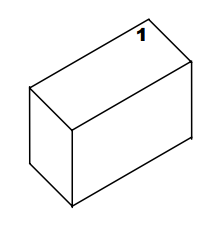
***NOTE:*** *Remember do not alter a user frame that is being used by your programs. injury to equipment or personnel may occur.*

1. Cursor to the frame to be set up and select F2 DETAIL.
2. Press F2 METHOD



1. Select Three Point
2. And the entry screen

will be displayed 🡪

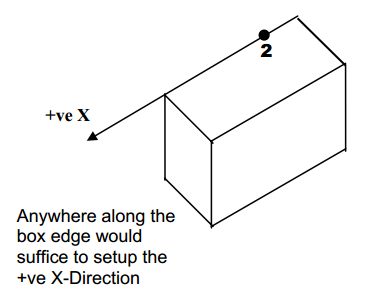


**Orient Origin Point**

1. Cursor to Orient Origin Point
2. Jog the Robot to a corner of

The box (1) on the diagram shown 🡪

1. Press SHIFT + RECORD (F5)



**X Direction Point**

1. Cursor to X Direction
2. To define the +ve X-Direction,

og the Robot to position 2 on

the box on the diagram shown 🡪

1. Press SHIFT + RECORD (F5)

**Y Direction Point**



1. Cursor to Y Direction
2. It is now necessary to define the

+ve X-Y plane. To do this jog the

Robot to anywhere on the box surface.

(shaded in the diagram) 🡪

1. Press SHIFT + RECORD (F5)

And the User Frame is now complete.

**Making a User Frame Active**

1. If a User frame is not made active, it cannot be used.
2. To make it active press the PREV key, and then SETIND F5.
3. The teach pendant will now ask which frame to make active, enter the number and press ENTER.
4. Go into the User Coordinate system using the COORD key, and the robot will now jog relative to the new User Frame that has just been set up.

***NOTE:*** *To get back to the default user frame, make user frame zero active.*

**Setting a User Frame active in a program**

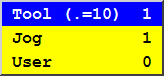
A user frame can be made active via the program screen.

1. To do this press INST
2. Select Offset/Frames (you may need to press 8 – next page – first)
3. Select UFRAME\_NUM = ...
4. Select Constant from the choices given
5. Finally enter the value of the user frame required.
6. Run this line using SHIFT + FWD to make the required frame active.

***REMEMBER*** *- Changing the active user frame changes the position of recorded position registers and can alter how a program runs. Injury or damage can occur if a user frame is changed or made active without the user fully understanding how user frames work.*

**Shortcut to setting a User Frame active**

1. Press and hold the SHIFT key
2. Press COORD



1. A small box will appear in the top right

hand corner of the teach pendant screen 🡪

1. Cursor to User and enter the frame number

you wish to use with the numeric keypad.

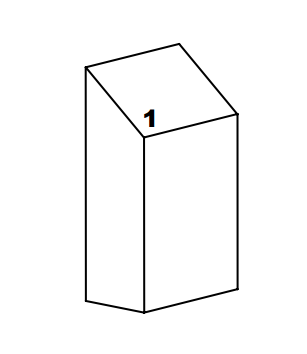
**JOG FRAMES**

*Setting up a jog frame for easy jogging*

A jog frame is a frame that can be created to allow for easy jogging in a plane that is different from the WORLD Frame. The X and Y directions are re-defined for JOGGING ONLY. Setting up and using a jog frame will have NO EFFECT during the running of a program.

**The Three Point Method**

1. Press MENUS and Select SETUP
2. Press F1 [TYPE] and select Frames
3. If jog frames are not displayed, press F3 [OTHER], and select jog frames.
4. If there are values other than zero in the frame you are using, press F4 (CLEAR)
5. Cursor to the frame to be set up and select F2 DETAIL.
6. Press F2 METHOD
7. Select Three Point
8. The entry screen will be displayed



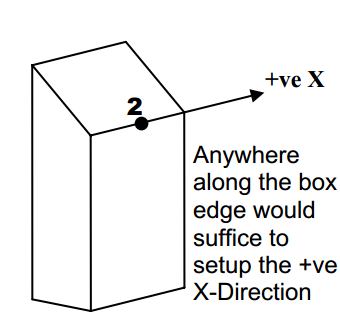
**Orient Origin Point**

1. Cursor to Orient Origin Point
2. Jog the Robot to a corner of the box (1)

on the diagram shown 🡪

1. Press SHIFT + RECORD (F5)

**X Direction Point**



1. Cursor to X Direction
2. To define the +ve X-Direction,

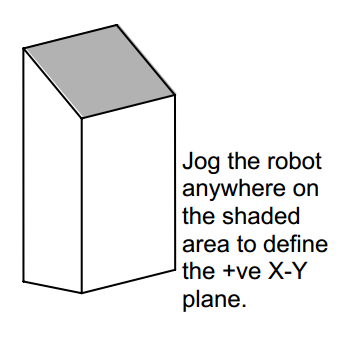
jog the Robot to position 2

on the box on the diagram shown 🡪

1. Press SHIFT + RECORD (F5)

**Y Direction Point**

1. Cursor to Y Direction



1. It is now necessary to define the +ve X-Y plane.

To do this jog the Robot to anywhere

on the box surface. (shaded in the diagram) 🡪

1. Press SHIFT + RECORD (F5)

The setup of a Jog Frame is now complete.

**Making a Jog Frame Active**

If the new Jog Frame is not made active, it cannot be used.

To make it active:

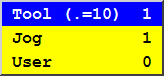
1. Press the PREV key, and then SETIND [F5].
2. The teach pendant will now ask which frame to make active, enter the number and press ENTER.

***NOTE****: A Jog Frame has NO EFFECT on a program. It is simply a way to jog easier in a plane other than the default.*

1. Go into the Jog Coordinate system using the COORD key, and the robot will now jog relative to the new Jog Frame that has just been set up.
2. To get back to the default Jog Frame, make a Jog Frame active that contains all zeroes.

**Shortcut to setting a Jog Frame active**

1. Press and hold the SHIFT key
2. Press COORD



1. A small box will appear in the top

right hand corner of the teach pendant screen 🡪

1. Cursor to Jog and enter the frame number you wish to use with the numeric keypad.

**UTILITIES**

**REFERENCE POSITION**

*How to record a reference position*

A Reference Position is a position in space where a Robot or Digital Output will automatically turn on. It exists independently of any recorded positions in a program.

For example, suppose you create a Reference Position that turns on an output whenever the robot is at the Home position. The Reference Position and the Home position are independent of each other.

If you later move the Home, you must move the Reference Position also.

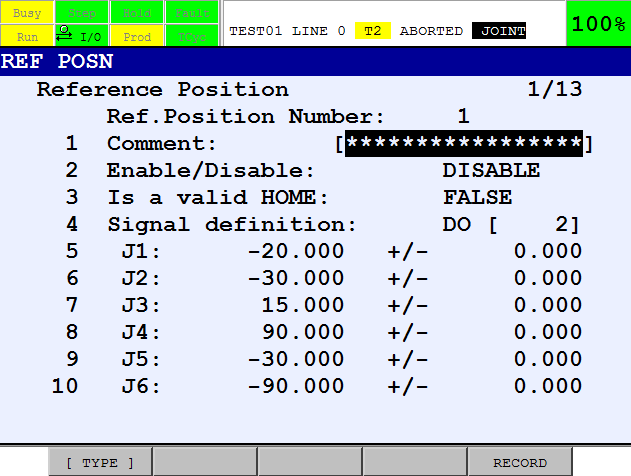
The Reference Position can be a single point in space OR it can be a zone determined by tolerance ranges.

**Setting up a Reference Position**

1. Create a program called Ref\_Pos.
2. Jog the robot to the position to

be used as a reference position.

1. Record the point



1. Press MENUS
2. Select SETUP
3. F1 [TYPE]
4. Select Ref Position
5. Curser to the Reference Position

you want to set up (1-10)

and press DETAIL [F3]

1. Press SHIFT + RECORD [F5],

and the position is recorded 🡪

**Enable/setup the Reference Position**

1. Cursor to line 2 and press ENABLE [F4]
2. Line 3, if set to Enable, this reference position will be checked as a Valid Home Position
3. Cursor to line 4. Signal Definition:
4. Select the output to be turned on at the Reference Position from the choices above the function keys. (DO or RO)
5. Cursor to the signal value and enter the required number via the numeric keypad.
6. The tolerance range of each joint can be changed in the right hand column. When the robot is within these position limits, the assigned digital output signal turns on.

(A tolerance of 0.5 to 1.0 per axis usually works well)

1. To check the reference position, go into the I/O screen (MENUS - I/O).
2. Jog the robot away from the reference position and out of the tolerance range.

- The selected output should turn off.

***NOTE****: If you want to get back to the exact position you recorded as a Reference Position, go into the program you created and execute it (move to).*

**MACROS**

***Record a macro program***

A macro instruction is very similar to a teach pendant program but has

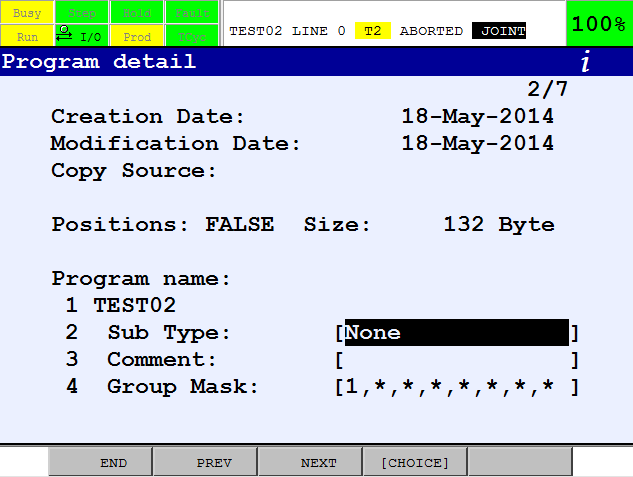
additional capabilities.

A macro instruction:

* + Can be called from within a teach pendant program
  + Can be started using the manual operation screen on the teach pendant
  + Can be started using a user key on the teach pendant
  + Can be started using inputs, DI, RI and UI.

**Creating a Macro Program**

1. Creating a teach pendant program, and go to Program details

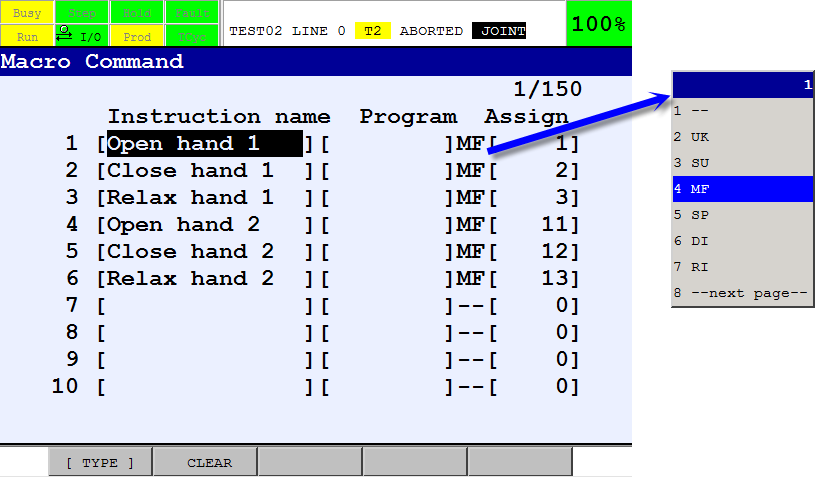


1. Curser down to Sub Type, and chose from None to Macro.

***Note****: If you wish to convert an existing teach pendant program into a macro program, press SELECT and then press [F2] DETAIL to get to this screen.*

**Setting a Macro Instruction**

1. Press the MENUS key and select 6. SETUP
2. Press [F1] TYPE and select Macro and the macro setup screen will be displayed.



1. Cursor to the macro you wish to setup and with the cursor in the ‘Instruction name’ column press ENTER
2. Use the alphanumeric keys to give a name that describes the macro.

When the name is complete, press ENTER.

1. Cursor over to the ‘program name’ column and press F4 CHOICE. A list of all the programs will be displayed. Select the macro name you wish to use.
2. Cursor over to the ‘assign’ column and press F4 CHOICE.
3. Select the option required and which number you require.

For example MF[1] for manual function key 1.

**Executing a Macro Using the Manual Functions Screen**

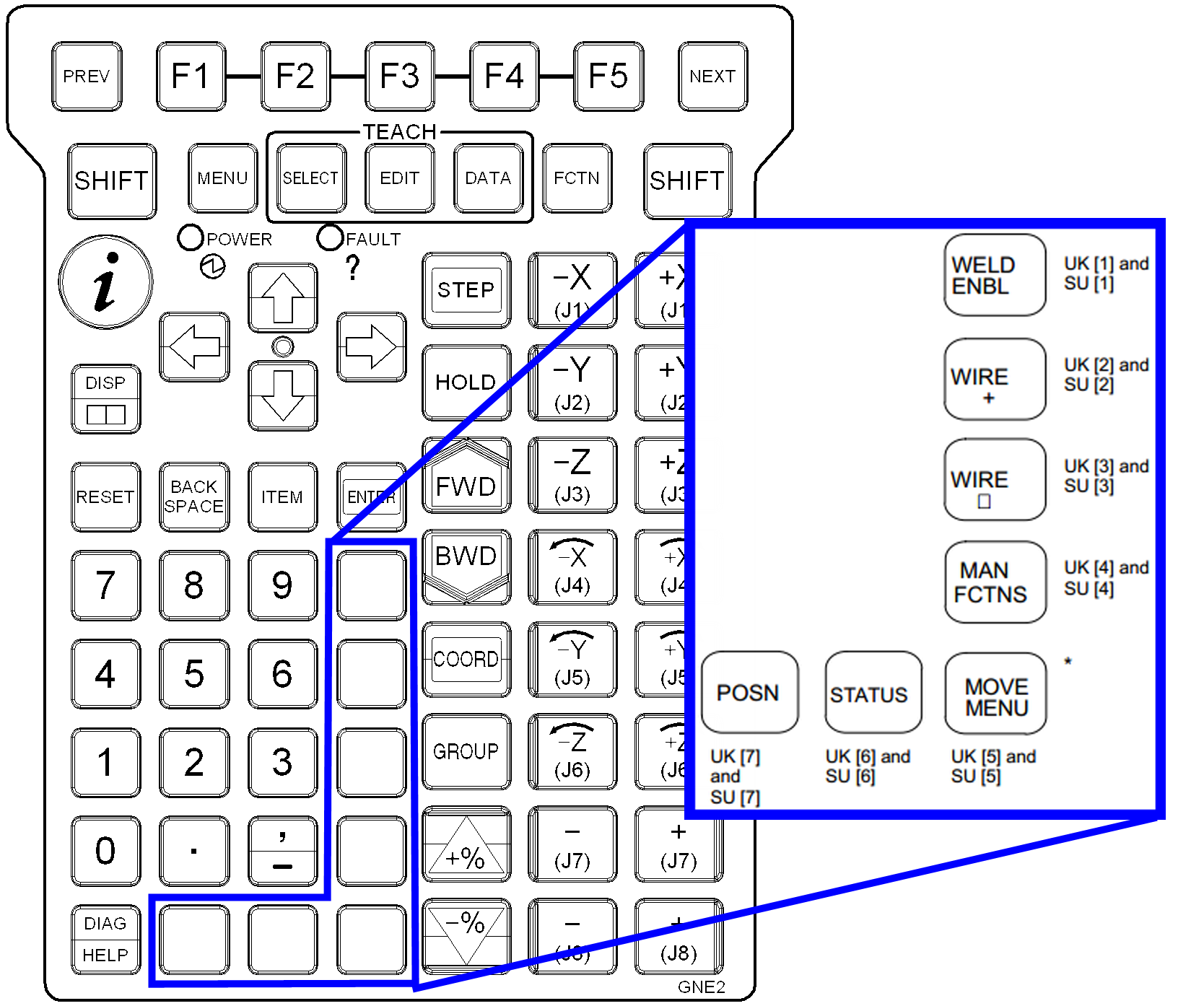
1. To display the Manual Functions screen:
2. Press the MENUS and Select 3. MANUAL FCTNS
3. The manual operation screen is displayed
4. Cursor to the Macro you wish to execute

***Warning****: The macro program is started in the next step, causing the robot to make a motion. Before executing the operation, the operator should check that no persons and no unnecessary equipment are in the work area. Otherwise, you could injure personnel or damage equipment.*

1. Press and hold the SHIFT + F3 EXEC

**Executing a Macro Using the Teach Pendant.**

Certain keys (user keys) on the teach pendant can be assigned to macro instructions.



UK = User Key

SU= SHIFT + User Key

Using the procedure above assign a macro to a user key (UK[1] to UK[7]) or (SU[1] to SU[7])

***Note****: A macro instruction that contains a motion group cannot be executed using ‘user keys’ UK[1] to UK[7]. It must be assigned instead to ‘Shift User Keys’ SU[1] to SU[7].*

To execute the macro, press the correct user key or press SHIFT + user key if SU was assigned.

**Running a Macro From a Teach Pendant Program**

You can call a macro from a teach pendant program by using the macro instruction on the INST Menu.

1. Insert a blank line in your program where you require the macro instruction to be.
2. Press F1 INST
3. Select MACRO. You may need to press 8---next page--- first.
4. Cursor through the list of macro programs until you find the one you require.

**Robot Payload Setting**

**Overview**

Robot Payload is the weight, or mass, of the robot end-of-arm tooling and workpiece.

If you have not set up the proper robot payload during software installation, or if you need to change the Robot Payload because you have changed end-of-arm tooling or the workpiece, you must set Robot Payload Data.

***Note****: It is important that the payload data used by the robot be as accurate as possible. The more accurate the values, the more effective features such as Collision Guard and Soft Float will be. Accurate values will also improve positional accuracy, cycle time, and general motion performance that are critical for today’s processing.*

You can define up to ten different payload schedules. You can then specify a payload schedule by using the payload setup screens and by using the payload teach pendant program instructions. You can set up payload schedules from the MOTION PERFORMANCE screen.

**Setting the Active Payload Schedule**

You can set the active payload schedule, either manually or in a teach pendant instruction.

**Manually Setting the Active Payload Schedule**

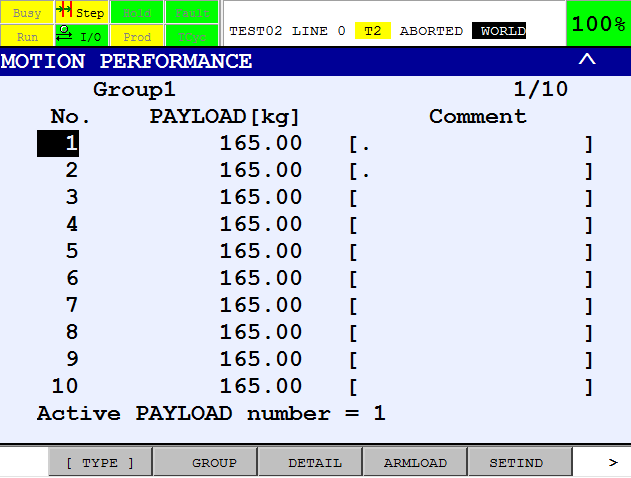
***Note****: You cannot update payload values when a program is running or if the active schedule number is the same as the schedule you want to modify.*

Conditions:

* SRDY is on.
* No motion commands have been issued.
* $PARAM\_GROUP[].$MOUNT\_ANGLE has not been set.
* Robot mastering/calibration has been performed.

**Procedure:**

1. Press MENU 🡪 0.Next 🡪 6.SYSTEM 🡪 6.Motion
2. You will see a screen similar to the following.



1. Press F5, SETIND
2. Type the number of the

payload schedule you

want and press ENTER.

**Using Teach Pendant Instructions to Set the Active Payload Schedule**

***Note****: Some applications and the Collision Guard function require the proper setting of payload information. If the payload changes during your application, you must use the PAYLOAD[x] instruction to select the appropriate payload schedule.*

***Note****: The PAYLOAD[x] instruction allows you to specify the payload schedule to use.*

*You can specify up to 10 different sets of payload information. Each set of payload information corresponds to a schedule number.*

**Procedure:**

1. Edit the teach pendant program in which you want to include PAYLOAD instructions.
2. In the F2, [INST] menu, Select Payload and press ENTER.
3. Select PAYLOAD[...] and press ENTER.
4. Type the value of the of the payload schedule:
5. Direct - type a schedule number and press ENTER.
6. Indirect - select INDIRECT, type a register number, and press ENTER.

PAYLOAD [1]

**Setup of Payload Schedules:**

There are two ways to set up the payload schedule parameters:

1. Manual entry
2. Automatic estimation using Payload Ident. (Option J669)

***Note****: Automatic robot payload estimation is not available for all robot models.*

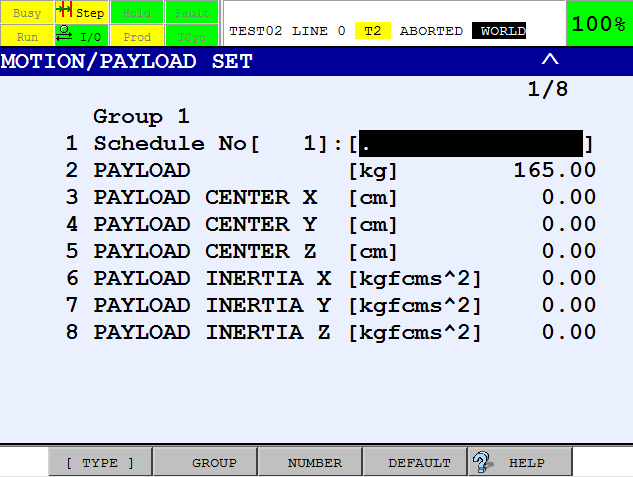
*(e.g. 4-axis robots)*

**Manually Setting Robot Payload Schedule**

**Procedure:**

1. Press MENU 🡪 0.Next 🡪 6.SYSTEM 🡪 6.Motion
2. Move the cursor the payload schedule you want and press F3, DETAIL

You will see a screen similar to the following.



1. To set the value, move the cursor to the items you want to set and set them as desired.

When you set payload, you set the values of several items related to payload.

Direction is relative to the robot tool frame with X, Y, Z, W, P & R set to zero,

and robot joint angles at the zero positions.

Refer to the software manual for more detailed information to set up correct Payload Data.

E.g. R-30iB\_HandlingTool\_operator\_manual\_[x.xx] pdf

**FILE MANAGEMENT**

*Manipulating Programs*

**PROGRAM HEADER**

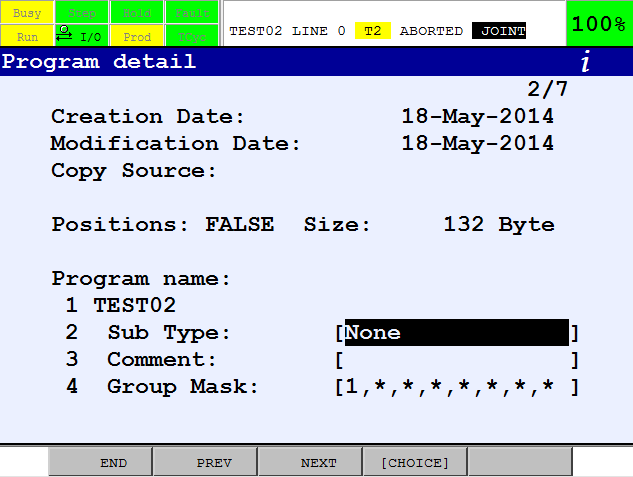
The program header contains information that identifies and classifies the program.

Program header information consists of:

* Creation Date
* Modification Date
* Copy Source
* Does the program contain positions?
* Program Size
* Program Name
* Sub Type
* Program Comment
* Group Mask
* Write Protection
* Ignore Pause

**Viewing Program Header Information:**

1. In the SELECT screen, curser to the program you want to view
2. If DETAIL is not above F2, press the NEXT key
3. Press F2 DETAIL



The screen will look like so:

*If the current program has been copied from another program, the name of the original program will be here.*

*Press F1, END when you have finished editing the PROGRAM HEADER screen.*

*This shows when the program was created and last modified.*

**Program Name**

1. The name of a program can be changed in the program header.
2. Cursor to Program name and press ENTER.
3. The alphabet will appear above the function keys
4. Enter the new name and press ENTER.

**Sub Type**

1. Cursor to Sub Type on the Program Header screen and press F4, CHOICE
2. There are two choices, None and Macro
3. Select None, if you wish to create a Teach Pendant Program

**Program Comment**

1. Cursor to comment on the Program Header screen and press ENTER
2. The letters of the alphabet will appear above the function keys and allow

you to enter a comment.

***NOTE****: A comment can contain any characters, including blank spaces.*

**Protecting a Program**

1. Cursor to Write protect
2. ON and OFF will appear above the function keys.

**DELETING A FILE**

1. You are in the SELECT screen
2. Place the cursor on the program you wish to delete
3. If DELETE is not above F3, press the NEXT key
4. Press F3 DELETE, and You will be asked

Delete OK ? YES NO

F4 F5

1. If you are sure you wish to delete the highlighted program, press F4 YES.

***NOTE****: There is no going back if you delete the wrong program!!*

*(Unless you have a backup of it)*

**COPYING A PROGRAM**

1. In the SELECT screen, cursor to the program you wish to copy
2. If copy is not above F1, press the NEXT key and Press F1 COPY
3. Enter a new name and press ENTER
4. Press ENTER again and the program will be copied.

**BACKING UP**

**Files**

When you back up files, you save it from controller memory to the default device so that you have a second copy of the files. You can back up program, system, application, diagnostic, and error log files to the default device using the FILE screen.

**Backing Up System and Program Files from the Teach Pendant.**

***NOTE****: If you are backing up files to a memory card or USB memory stick device, make sure that the memory card or memory stick is installed properly.*

**Procedure**

Set the default device to the device you want,

e.g. UT1 (USB on TP) or UD1 (USB on controller front)

1. Using the teach pendant,
2. Press MENU and select 7.FILE
3. Press F1, [TYPE] and select 1. File
4. Press F5, [UTIL] and Select Set Device.
5. Move the cursor to the device you want and press ENTER.
6. Press F2, [DIR].
7. Select 1. \*.\*
8. Choose a folder for files to be backed up:

* Create a folder, press F5 [UTIL] and select 4. Make Dir.

(remember to enable the teach pendant)

* Or navigate to your folder, already on your device

1. Press F4, [BACKUP]
2. Select an item that you want to back up from and press ENTER.

*Typically, an “All Of Above” backup is performed.*

1. Choose the backup operation that you would like to perform from and press ENTER.

**Loading and Restoring Files to Controller Memory**

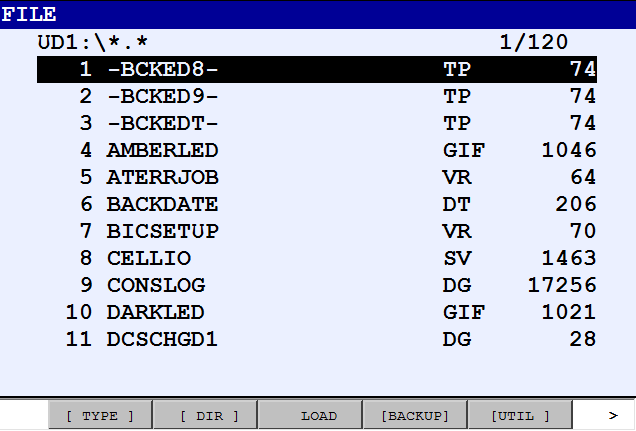
Loading and restoring files allows you to load a file and all of its relevant data from memory card (MC) or USB memory stick into controller memory.

**Loading Files Using the FILE Menu**

If you are loading files from memory card or USB memory stick, make sure that the memory card or memory stick is installed properly

**Procedure**

1. Press MENU and select 7. FILE
2. Press F1, [TYPE] and select 1. File
3. Press F5, [UTIL] and select Set Device e.g. UD1 (USB on controller front)
4. Move the cursor to the device you want and press ENTER.
5. Press F2, [DIR] and Select 1. \*.\*



1. Navigate to the folder with files.

To load a single file:

1. Move the cursor to the name

of the file you want to load

and press F3, LOAD.

1. Load the file(s):
2. To load the file(s) you selected, Press F4, YES.
3. If you do not want to load the file(s) you selected, press F5, NO.

**Restoring BACKUP Files via Controlled Start FILE Menu**

**Procedure**

1. Perform a Controlled start.
2. Press and hold, **PREV** and **NEXT** keys on the teach pendant,

and cycle the main power.

1. After a few seconds, you can then release the PREV and NEXT keys.
2. When you see the Configuration Menu, choose 3. Controlled start and press ENTER.

When booting up in Controlled start is complete and ready,

* 1. Press MENU and select 5. FILE
  2. Press F5, [UTIL] and select Set Device.
  3. Move the cursor to the device you want and press ENTER.
  4. Press F4 [RESTORE]
  5. Select 1-6 and start the file restore function.
  6. When done with the restore, press FCTN and select 1. Start Cold.

**IMAGE BACKUP**

The Image Backup feature allows you to make an image of the F-ROM and S-RAM controller memory modules. The image is stored in a number of files (\*.img) on a selected destination device.

Image Backup is available from the FILE menu when the system is in Cold start or Controlled start mode. After Image Backup is selected from the menu, the actual backup occurs the next time the controller is powered up.

The image created can later be used to restore the controller memory. The restore is also done during power up.

If **F1** and **F5** are pressed while the controller is powering up, a menu is displayed that allows you to restore a previously saved image.

**Performing an Image Backup**

**Procedure**

* Insert a memory card, a USB memory stick or setup TFTP.

***Note:*** *The memory card interface is not available on the R-30iB Mate Controller.*

1. Press MENU and select 7. FILE
2. If using MC: or UD1: select the device and sub-directory.

(If using TFTP: you will be prompted later on)

1. Enable the Teach Pendant
2. Press F4, [BACKUP] and chose Image Backup
3. Cycle Power is needed to write the Image Backup to your selected device.

**Restoring an Image Backup**

**Procedure**

1. Press **F1** and **F5** while

\*\*\*\*\*\*\* BMON MENU \*\*\*\*\*\*

1. Configuration menu

2. All software installation(MC:)

3. INIT start

**4. Controller backup/restore**

5. Hardware diagnosis

6. Maintenance

7. All software installation(Ethernet)

8. All software installation(UD1:)

turning on the power.

The BMON MENU

will be displayed 🡪

1. Select

4.Controller backup/restore

and press ENTER.

\*\*\*\*\*\*\* BACKUP/RESTORE MENU \*\*\*\*\*\*

0. Return to MAIN menu

1. Emergency Backup

2. Backup controller as Images

**3. Restore Controller Images**

4. Bootstrap to CFG MENU

1. On next screen, Select

3.Restore Controller Images

and press ENTER.

\*\* RESTORE Controller Images \*\*\*\*

1. Memory card(MC:)

2. Ethernet(TFTP:)

3. USB(UD1:)

4. USB(UT1:)

1. On next screen, Select e.g.

1.Memory card (MC:)

or 3.USB (UD1:)

and press ENTER.

1. On next screen, confirm that

\*\*\*\*\* RESTORE Controller Images \*\*\*\*\*

Current module size:

FROM: 32Mb

SRAM: 3Mb

CAUTION: You SHOULD have image files

from the same size of FROM/SRAM.

If you don’t, this operation causes

fatal damage to this controller.

Are you ready ? [Y=1/N=else] :

You want to proceed, by

Pressing 1 and ENTER

***Caution****: This operation erases all of SRAM and FROM.*

*Be sure you are prepared to erase all of SRAM and FROM before you execute*