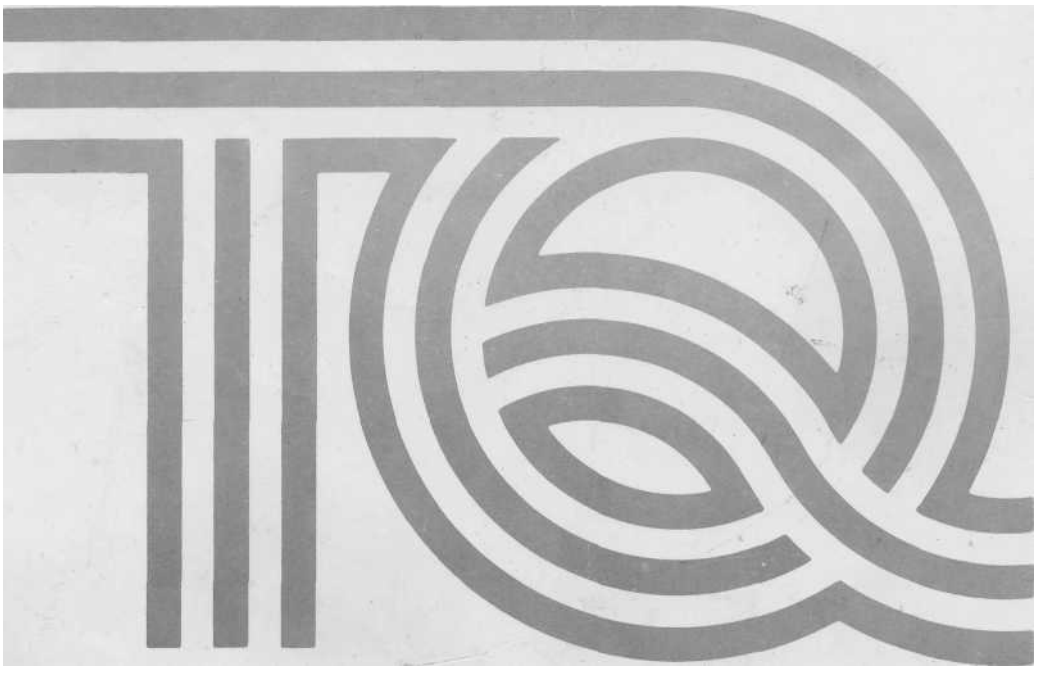


TecQuipment

MA2000 ROBOT
Operators Manual

The equipment described in this manual is
manufactured and distributed by the
T E C Q U I P M E N T
Group of Companies

Suppliers of Technological Laboratory
Equipment designed for Teaching

BONSALL STREET, LONG EATON, NOTTINGHAM, NG10 2AN, ENGLAND
TELEPHONE: LONG EATON 722611 TELEX: 377828 FAX: 731520

VEK/RR/1187/i

Whilst all due care has been taken to ensure that the contents of this manual are accurate and up to date, no responsibility, direct or consequential, can be accepted for any errors or omissions.

A Packing Contents List is supplied with the equipment and it is recommended that the customer checks carefully the contents of the package(s) against the list ensuring that no items are missed.

Should there be any items missing, the customer should contact the local TecQuipment agent or TecQuipment direct.

C O N T E N T S

SAFETY WITH ROBOTS

(a) Safety for Humans

Robots seem to be very clever, responding eagerly to the instructions of their operators, but in reality they are merely machines performing mechanical functions. As such they are sublimely unaware of the presence of human heads and hands and eyes, and have no instinct or common sense to warn them of imminent danger to life or limb or sight.

In any collision between a robot and a human, the human is the most likely to suffer permanent injury, and the human is the only one who will feel pain.

Keep out of the way of moving robots. Stay outside the operating envelope while the robot is active. Make sure other people do the same.

(b) Safety for Sleeping Robots

This is not to say that robots cannot be damaged by humans. Robots are particularly vulnerable when immobile - when being transported or unpacked or set up, and once they have been set up they are likely to have their joints sprained if they are moved into "limits" too violently.

(c) Safety for Robots under Tuition

Although the method of robot teaching called 'Lead-by-the-nose' calls to mind the image of a tiny farmboy controlling a massive bull by means of a ring through the bull's nose, this metaphor has its limitations. When a robot is being taught by this method it has no motive power of its own, and unlike the bull it cannot even stand upright. Support must be given to the shoulder and elbow, even if their positions are not being altered during the present move. Even if you are only moving the "end effector" you will need to take the weight of the rest of the robot.

(d) Safety for Robots in Motion

Robots are also capable of damaging themselves. If a moving robot collides with an inanimate object it is obviously likely to suffer. Be kind to your robot friend and don't drive him into walls or control boxes or smash him into his own base board. He wont complain, he will just break.

This manual is intended for use with computers using BBC BASIC. It is based on the IBM using BBC BASIC 86. Notes have been included for BBC computer users wherever the operations differ from the IBM. Whatever computer is used, it is always advisable to use the tutor text when starting and to READ THE PROMPTS.

1. INTRODUCTION

The MA2000 robot is designed to give a full range of facilities at low cost. The designers have made every effort to simplify the operating procedures, but, even so, the MA2000 is a complex piece of equipment. Driving it is every bit as difficult as driving a motor car, and requires just as much patience, persistence and skill.

The following guidelines MUST BE READ and UNDERSTOOD by all who intend to operate the robot.

UNDER NO CIRCUMSTANCES SHOULD THE ROBOT BE OPERATED WITHOUT PROPER INSTRUCTION OR UNDERSTANDING OF THE FOLLOWING NOTES.

NOTE: The robot has a reach of 500mm and a "dead lift" of 1kg. This means it will support lkg at the end effector attachment point (i.e. 480mm radius). When the robot is lifting and moving weights, a realistic maximum is 500gm.

Concept and Terminology

Before you switch on your new robot, it is worthwhile checking that we both speak the same language (especially if you are new to robotics), and that you understand the principle of the robot operation.

We'll start on the basics:

Mechanically, the robot consists of an arm (this being what most people refer to as "THE ROBOT") with 6 joints in similar positions as on a human arm.

Refer to figure 1 and your own robot and you will see that first of all the robot can pivot on its base about a vertical axis; this is the WAIST motion. Next is the SHOULDER motion with a pivot point just above the waist motion. Halfway along the robot is the ELBOW joint. These three motions, the major axes, do the majority of the work in a typical robot cycle which explains the chunkiness of these motors compared with the remaining three.

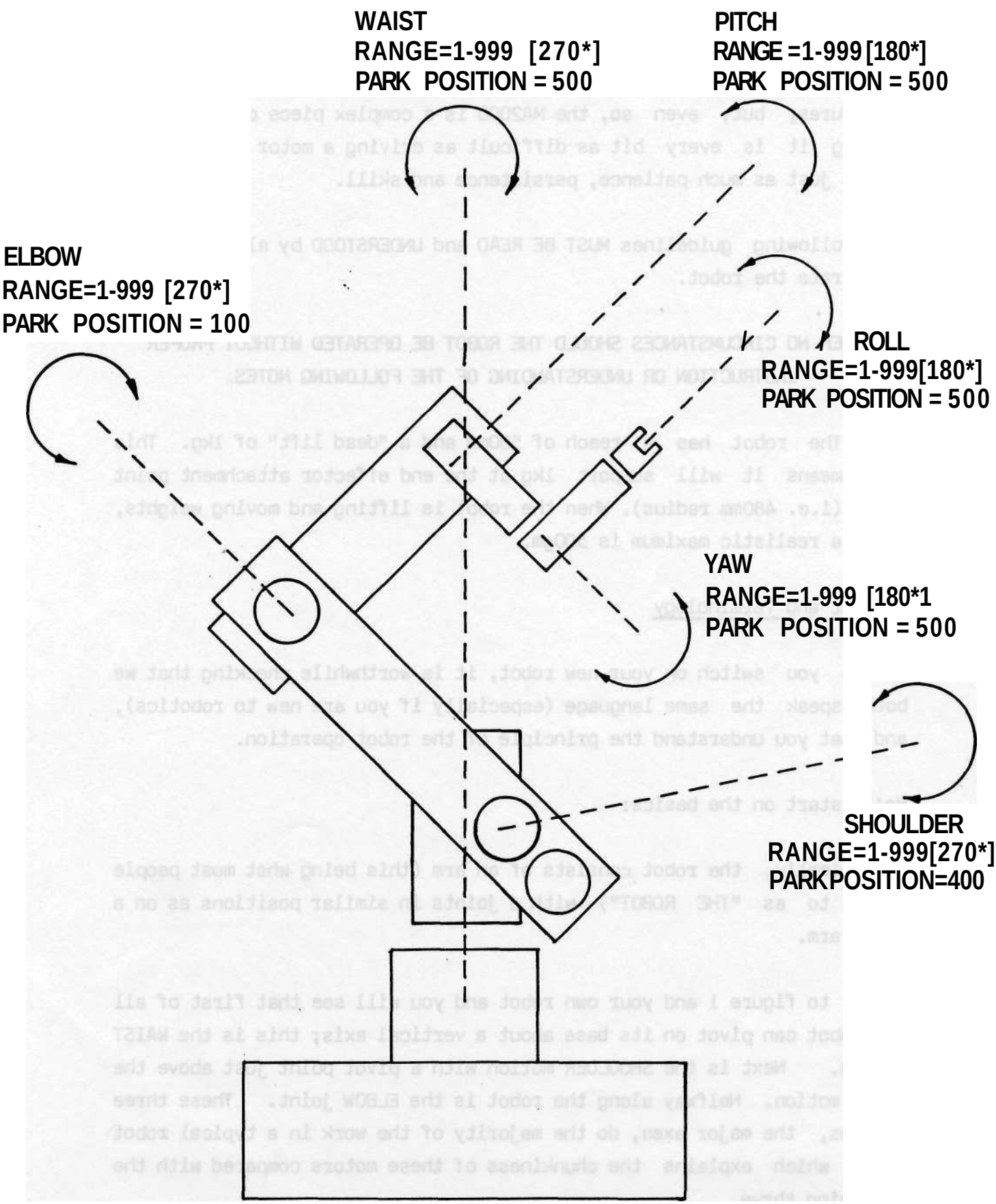


FIGURE 1 :- ROBOT MAJOR AND MINOR AXES SHOWING PARK POSITION

The final three motions, the minor axes, at the end of the robot, (which you may consider as being a hand; the "proper" name in robot circles is "end effector"), are the PITCH, which moves vertically up and down, the YAW, which gives a sideways swivel, and the ROLL which imitates, for example, a screw driver movement.

The range of movement on the major axes is 270° and 180° on the minor axes. This means that the robot can reach most points inside an imaginary hemisphere with a 500mm radius, centred on the shoulder joint.

Turning to the electrical components (housed in the large box) your money has bought you a MOTOR CONTROLLER, and an interface to the HOST COMPUTER. Finally, there is the KEYPAD which is the calculator-like box which plugs, via its long cable, into the front panel of the motor controller.

The keypad is the channel of communication with the robot. With it you can, for example, say in which direction you want the robot to move and how fast. The computer translates the keys that you press into signals to the motor controller which then provides the power to the motors to move them accordingly. Having moved the robot to a position, the computer can then remember the position and allow you to program the next.

Once you are satisfied that you have a sequence of moves that will perform your task, then the sequence can be initiated via one push button on the keypad. The computer will then step through the stored sequence sending the positions to the motor controller which in turn provides the motor power to move the robot.

Having introduced you to the basic concept of the robot operation and some of the terms, once your robot is installed you can move straight on to operating the robot by following the book step-by-step.

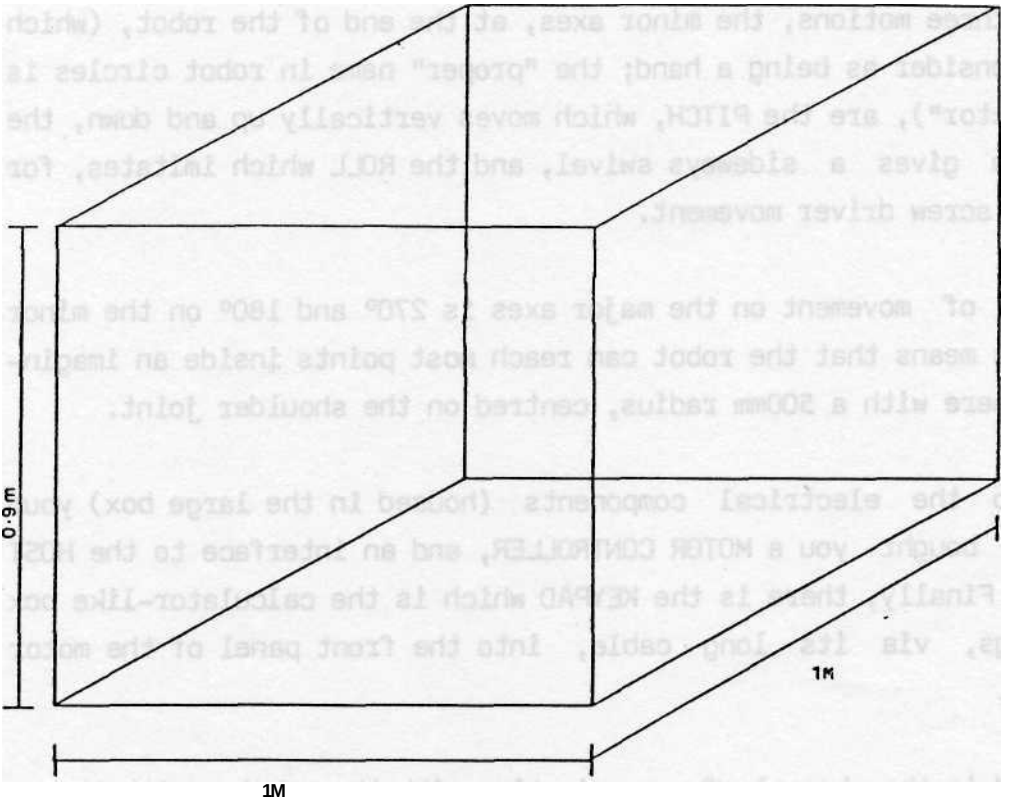


FIG 2.2.1

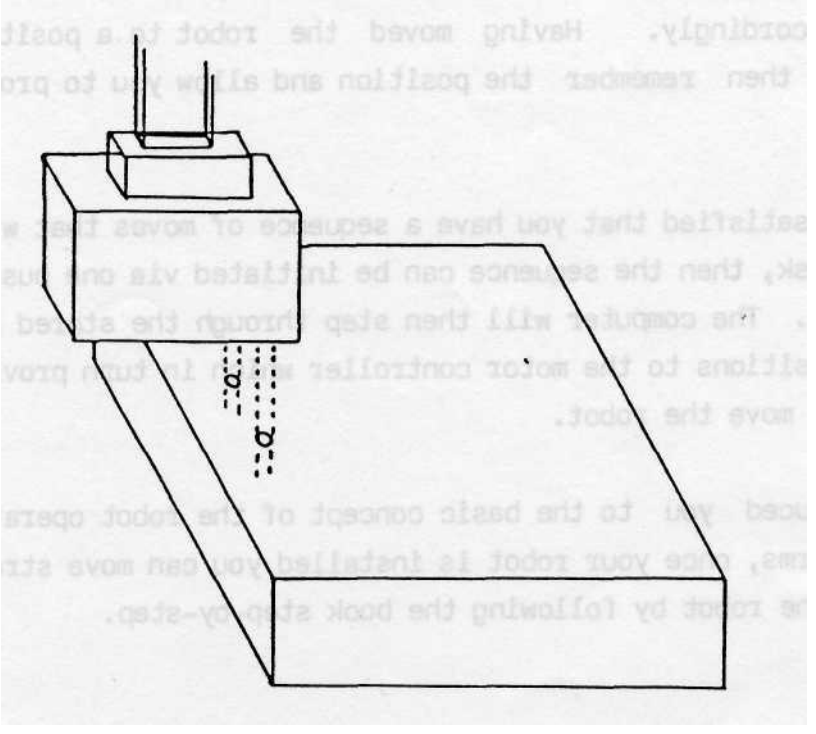


FIG 2.2.2

2. UNPACKING AND INSTALLATION

It is important that the following unpacking and installation guidelines are carefully read and understood.

The robot arm could easily be damaged if it is handled incorrectly.

2.1 Unpacking

Suitcase-type container

(1) Check that the robot arm case is resting with its lid facing upwards.

(2) Unlock the case clips and remove the lid.

(3) Holding the robot at its yellow base and elbow gently lift it out of the case.

It is important that the weight of the robot arm is taken when moving it between case and installation site.

(4) Keeping the arm vertical, move the robot to its installation site.

(5) Set the robot base down on its flat surface.

Failure to place the robot on a flat surface could damage the robot.

Conventional container

(1) Ensure that the case is resting with its lid uppermost.

(2) Remove the screws securing the lid; and lift the lid vertically away from the container.

(3) On the side panel secured only by screws, remove the screws and lift away the panel.

(4) Remove the controller and stow in a safe place.

(5) Remove the packing notes and ancillary equipment, stowed below the shelf housing the robot and controller.

(6) Remove the four M6 bolts securing the robot base to the plywood shelf.

(7) On the wooden clamp holding the robot forearm secure, slacken off the bottom M10 bolt and remove the top M10 bolt.

(8) Remove any ties securing the cables to the plywood shelf.

(9) Gently lift the forearm to the vertical and use the other hand to grasp the waist motor; lift the robot away from the packing case.

(10) Set the robot base down on a flat surface.

2.2 Installation

Choosing your installation site

When choosing a site for the robot, it is important to consider the operational area involved. The site must certainly be flat and free from any immediate obstruction.

Although the actual robot operating space should be depicted by a hemisphere, it is more practical to use a cubic representation. We suggest that a cube of dimension 1m x lm x 0.9m will be the necessary space needed to ensure complete operating freedom of the robot.

Fig 2.2.1 illustrates the robot operating space.

Having satisfied the siting requirements, the robot must now be mounted on a suitable board. The robot suitcase-type container can be used as the mounting board, otherwise a suitable mounting board will have to be made.

The first set of instructions are relative to those using the robot case as the mounting board.

Transforming the robot case into a robot base

(Suitcase-type container only)

1. Place the two halves of the case side by side.
   The set of drilled holes correspond to the mounting holes in the underside of the yellow robot base. The cable side of the yellow robot base should be mounted on the case base and the other side on the case lid.
   You will find the remaining instruction impractical without the aid of a colleague. THEREFORE HELP SHOULD NOW BE SOUGHT!

2. Use one person to hold the robot horizontally, again taking care to take the weight of the robot arm. The other person is then free to line up the holes in the case base to those in the robot base (see fig 2.2.2).

3. Using the M6 screws included in the MA2000 package, secure the robot base to the case base.

4. Having checked that the robot is firmly fixed to the case base, now line up the remaining two holes in the robot base to those in the case lid.

5. Secure the robot base to the case lid using the M6 screws provided.

6. Check that all the screws are tightly finished.

Now carefully place the entire unit in your chosen installation site. Keep the robot arm in a vertical position when it is left unconnected to the computer/controller.

Designing and installing a personal mounting board

(Robots in conventional container)

We recommend that the board should be no smaller than 1.2m^2. A single piece of plywood 12mm thick or greater will constitute an ideal board.

Having found a suitable board, the next procedure is to mark out the hole positions.

1. Lay the mat provided with the robot (Fig 2.2.3(a)) on to the plywood board.

2. The robot base area marked in the centre of the mat also has the hole positions marked for the securing bolts. Check that these dimensions agree with those shown on Fig 2.2.3(b).

3. Drill the four holes as indicated using a 6.5mm drill and remove any burrs.

4. Counter sink the holes on one side of the board. This is done to retain the flatness of the board.

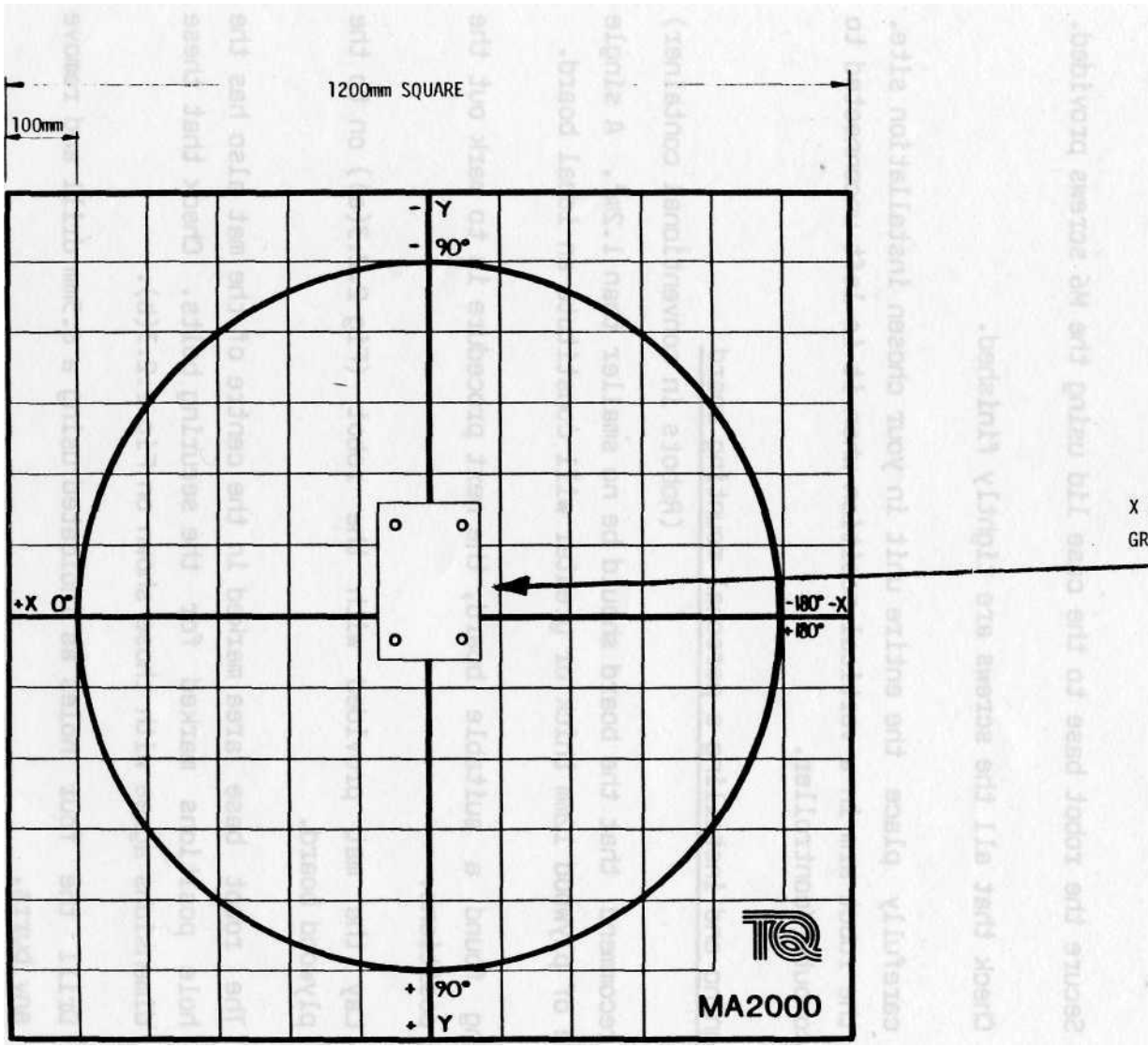


Fig 2.2.3(a) TQ Robot Mat

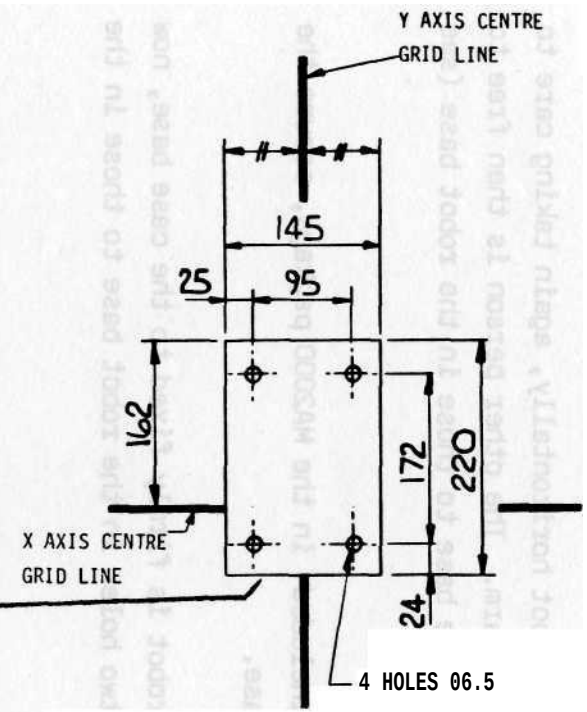


Fig 2.2.3(b) Drilling Detail on Robot Mat

5. Check that the M6 screws included in the MA2000 package fit the drilled holes.

WE ARE NOW READY TO MOUNT THE ROBOT BASE ONTO THE PERSONAL MOUNTING BOARD.

6. Place two tables approximately 0.4m apart.

7. Rest the board on the two facing edges of the tables, making sure that the countersunk side faces downward.

8. Looking at the underside of the board, the countersunk holes should be accessible in the space between the tables. If all, or some, of the holes are hidden by the board/table overlap, gently move the board until all four holes are visible.

9. Ensure that the mat is in position on the board and place the robot base over the drilled mounting holes in the topside of the board.

10. The vertical position of the robot arm must be maintained by a colleague.

11. Using the M6 screws provided, fix the robot base to the board.

12. Check that all screws are tightly fastened and place the entire unit in its installation site.

2.3 Making the Connections

First check that there is a mains supply near to the installation site. It is likely that a mains adaptor will be needed because several items of equipment require mains supply. The equipment should then be installed in the user work area free from robot intervention.

It is advisable to keep the 'emergency stop' button, on the controller face panel, within easy reach.

2.3.1 Installation of the IBM Interface, MA2000w

1. Tools Required

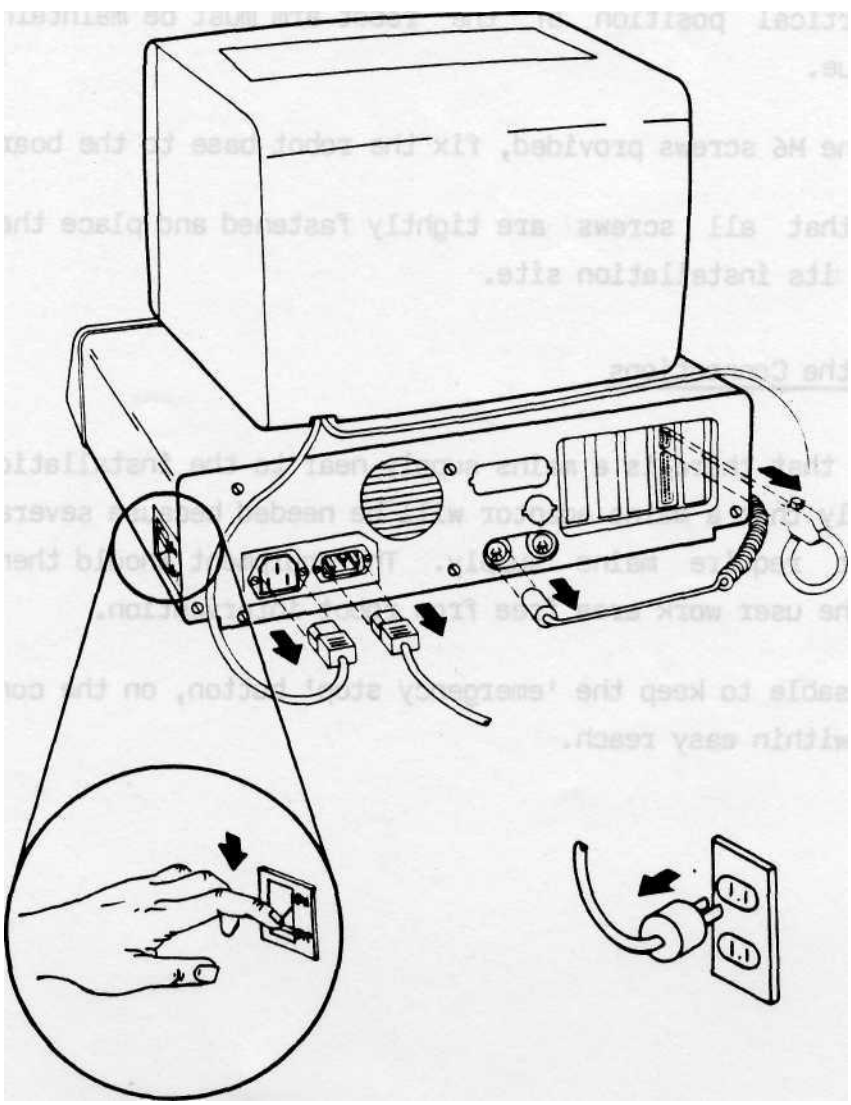
Medium flat blade screwdriver.

2. Preliminary Steps

Position system unit switch to OFF.

Position any external option power switches to OFF. (Printer, monitor, etc)

Unplug system unit and all other options.

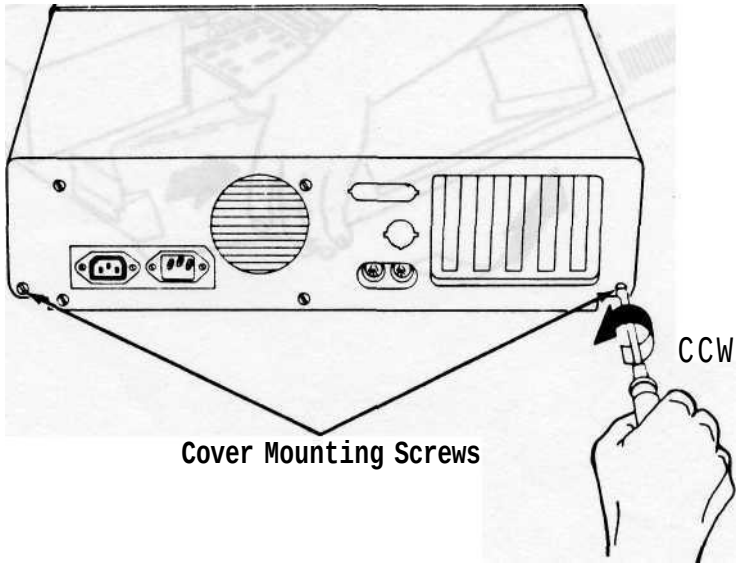


3. Installation

Set keyboard and other options away from the work area.

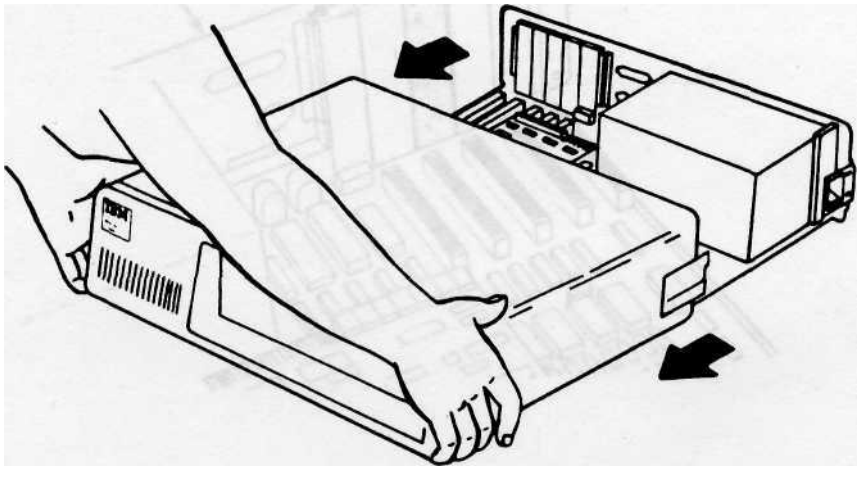
Position system unit to allow rear access.

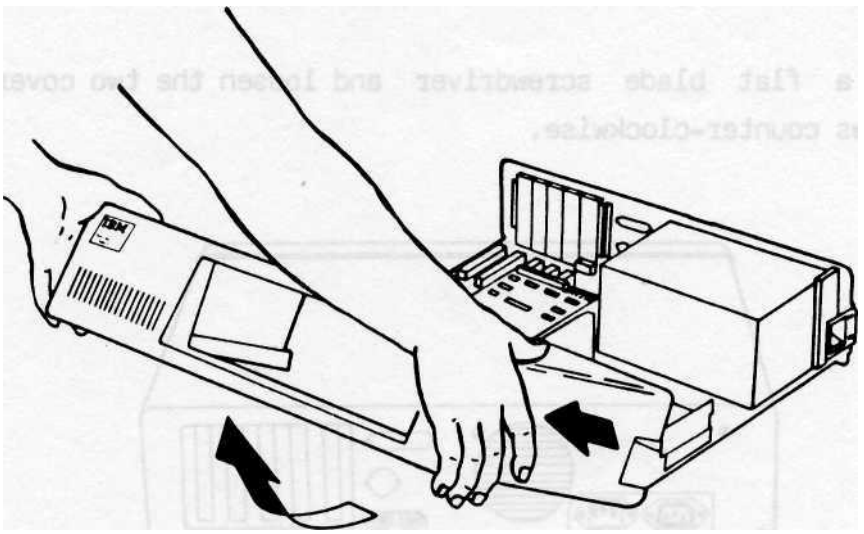
Use a flat blade screwdriver and loosen the two cover mounting screws counter-clockwise.



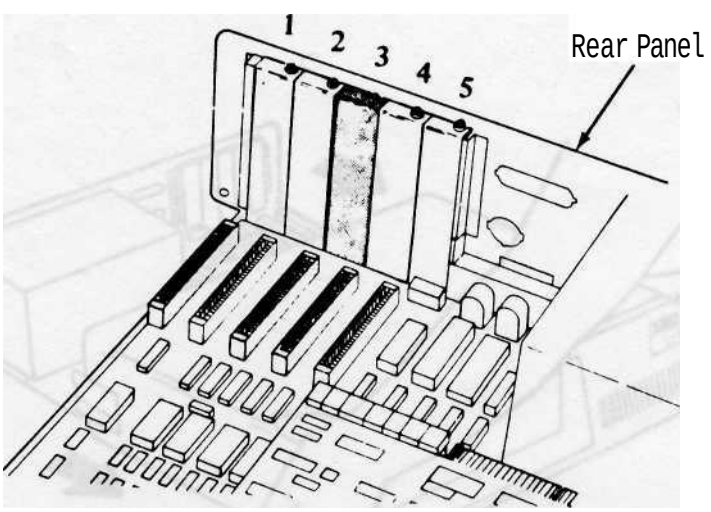
Cover Mounting Screws

Carefully slide the cover away from the rear and towards the front, until it will go no further. Tilt the cover up and remove it from the base, and set aside.

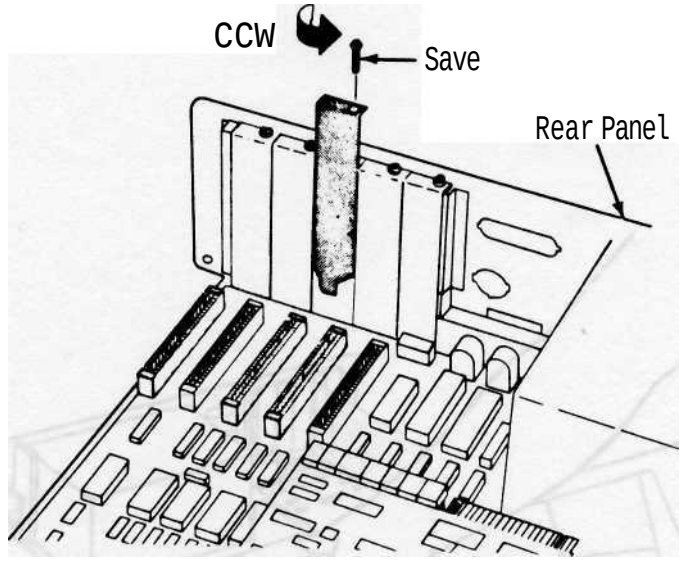




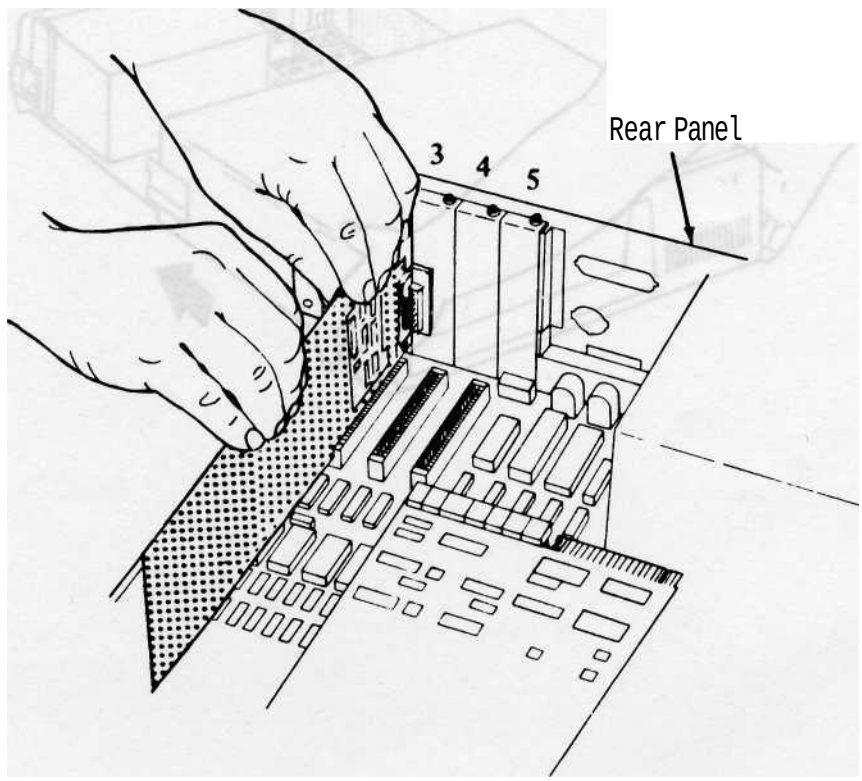
Look at the inside left of the system unit. There are five slots. It is recommended that slot 3 is used.



Remove the screw that holds the system expansion slot cover in place.

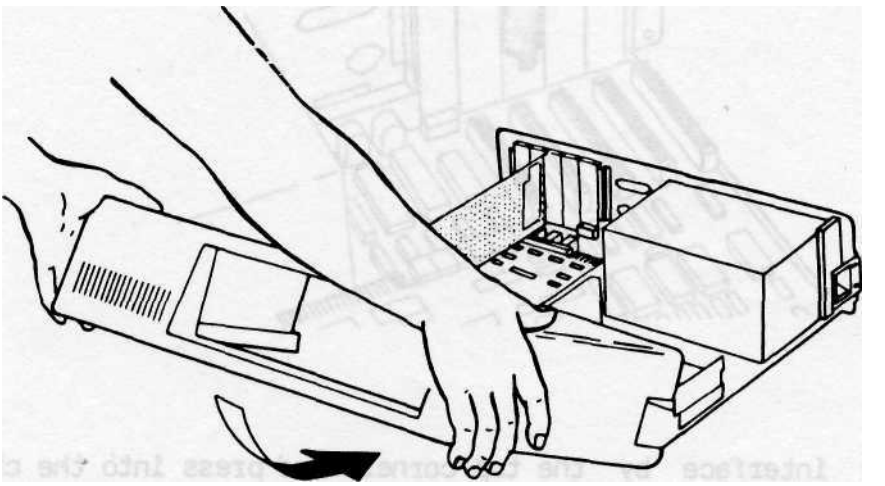


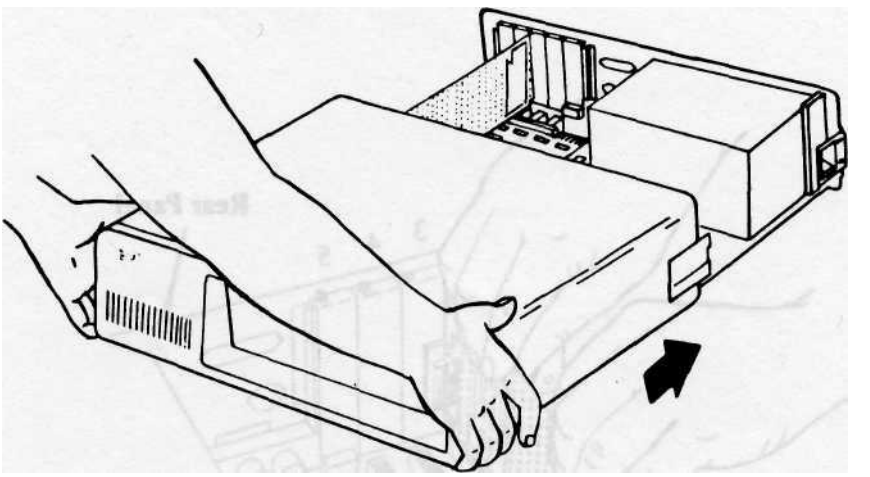
Hold the interface by the top corners and press into the chosen slot.



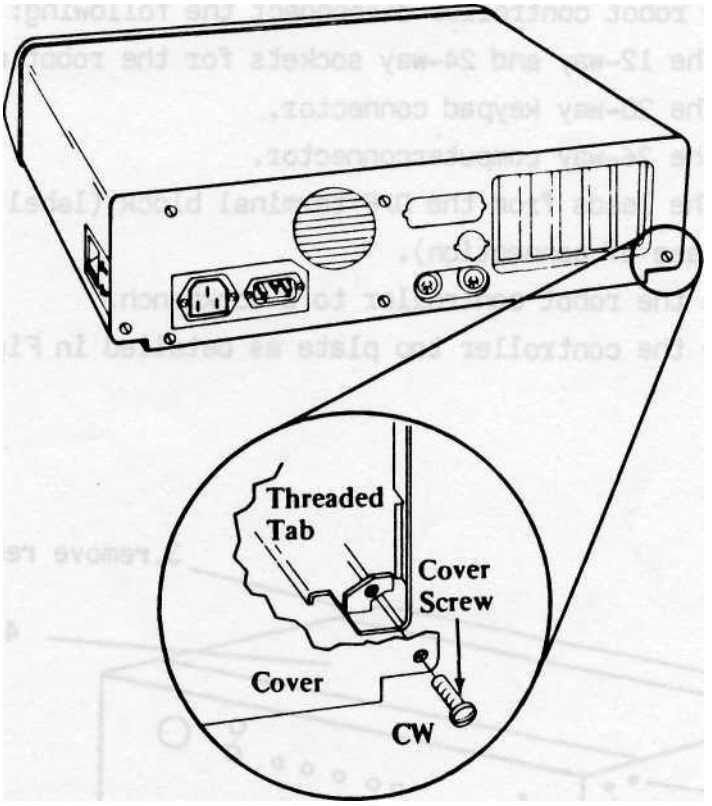
If you have any other options to install, do this now.

Replace cover on system unit in the reverse of removal.





When the cover is all the way to the rear, align the screw with the thread tabs and tighten.



The system may now be re-cabled and verified as explained in the appropriate IBM manual.

2.3.2 Conversion of Robot Controller

These instructions need only be used if the system used a BBC computer which is being replaced with an IBM or compatible machine.

1. Tools Required

Medium flat bladed screwdriver.
Soldering iron.

2. On the later versions of the controller a switch is fitted on the base adjacent to the centre of the front panel; move this to position B. If the switch is not fitted then proceed as follows:

(a) Disconnect the robot controller from the mains supply.
(b) On the robot controller disconnect the following:
 (i) The 12-way and 24-way sockets for the robot arm.
 (ii) The 20-way keypad connector,
 (iii) The 26-way computerconnector.
 (iv) The leads from the 0/P terminal block (label each lead for ease of connection).
(c) Remove the robot controller to a workbench.
(d) Remove the controller top plate as detailed in Fig 2.3.2.1.

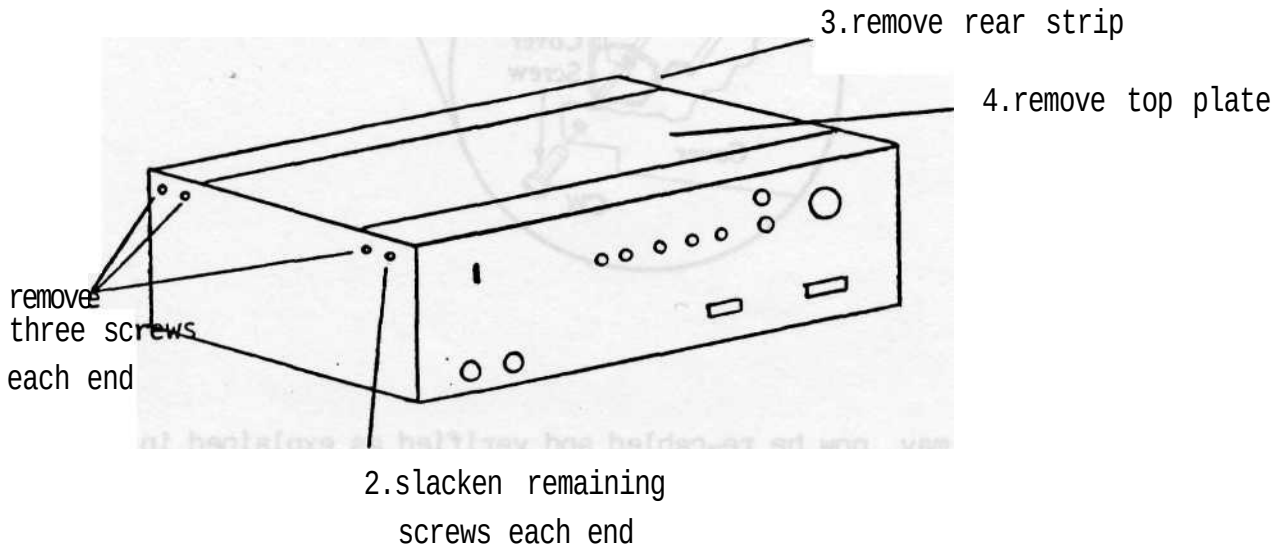


Fig 2.3.2.1 Removal of Controller Top Plate

(e) Locate the controller PCB 820-41349.
(f) With reference to Fig 2.3.2.2; locate the wire link between 9 and 10, unsolder the link and reconnect between 10 and LK.
(g) Refit the controller top plate.

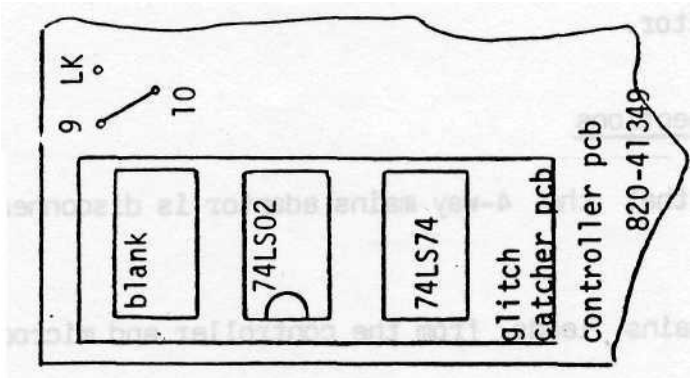


Fig 2.3.2.2 Controller PCB 820-413A9 Wired for BBC Operation

(h) Refit the robot controller in its installed position and reconnect the plugs and plugs and leads disconnected in para 2.b.

2.3.3 Computer Connection

Connect one end of the 26-way ribbon connector to the 26-way plug on the MA2000w interface card for the IBM and to the 1 MHz Bus for the BBC.

Always ensure that the ribbon cable connections are made with the ribbon stripe going to pin 1. Note: Pin 1 is marked with an arrow on the connector body.

2.3.4 Controller Connections

Always make sure that ribbon cable connections are made with the ribbon stripe going to PIN 1 in the connector.

1. Connect the keypad ribbon cable plug to the connector labelled KEY-PAD to the front face panel of the controller.

2. Connect the free end of the computer ribbon cable to the connector labelled COMPUTER.

3. Connect the two plugs at the end of the screened robot cable into the back of the controller. One of the plugs should be 12-way, the other 24-way. Because of the physical dimensions of the controller connectors, there should be no ambiguity as to which plug fits which connector.

2.3.5 Other Connections

1. Make sure that the 4-way mains adaptor is disconnected from the mains.

2. Plug the mains leads, from the controller and microcomputer into the adaptor.

3. Set the equipment power switches to ON.

4. Connect the 4-way mains adaptor to the mains but do not switch on.

5. Check your connections against Fig 2.3.2.3.

6. Connect the plastic air line between the robot air inlet and a clean dry regulated air supply (approximately 3 bar).

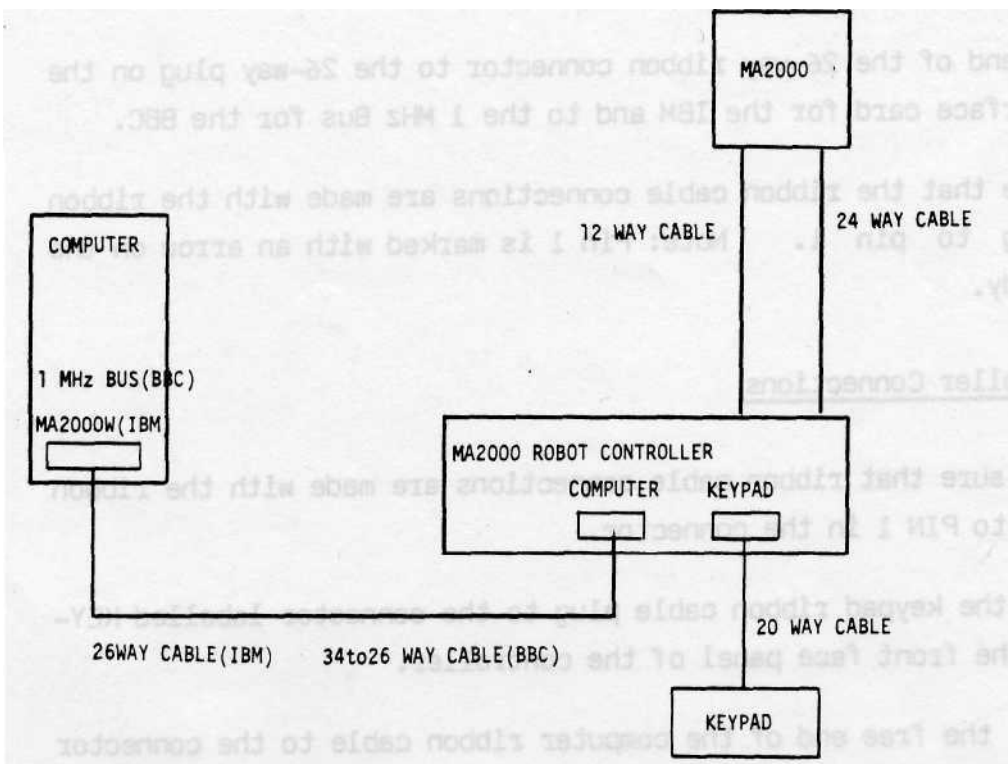


Fig 2.3.2.3

2.4 Powering-up the MA2000 System

Note: If the robot is going to be operated on its own, now is a good time to connect in the decision box MA2000p (see enclosed instruction sheet).

Switch the mains switch ON. The controller LIMITS light may light. Both the red EMERGENCY STOP and COMPUTER FAULT indicators will light. We are now ready to load the operating system.

2.5 Loading the Operating System

Note: paragraphs 1-5 are for IBM users and paragraphs 6 and 7 for BBC users.

1. You will require a disc or discs containing the disc operating system DOS and BBC BASIC(86) as well as the MA2000 robot software disc.

2. Remove the protective card from the mini-diskette.

   You may need to first unlock the diskette by turning the locking key anti-clockwise (see fig 2.5.1).

3. Take the disc operating system (DOS) disc out of its protective cover and place it in Drive A, ensuring that the disc slot enters first and label face upwards (see fig 2.5.1).

   Remember to lock the diskette by turning the locking key clockwise.

4. Switch ON the computer. (Some systems are fitted with RESET, this may also be used). The system will now self-check and load DOS.

5. If BBC BASIC 86 has not been installed on your MA2000 disc then carry out the following:
   (a) Remove the DOS disc from drive A and fit the MA2000 disc.
   (b) Type in "INSTALL" and follow the screen prompts.

   If BBC BASIC 86 is resident on your disc; once DOS has loaded, remove the DOS disc from drive A and fit the MA2000 disc. Type in MA2000 and the system will auto boot.

6. BBC users need only insert the MA2000 disc into the disc drive and press <SHIFT><BREAK> for the system to boot.

DSCLI"(CAT) " + seqnumber$ + "

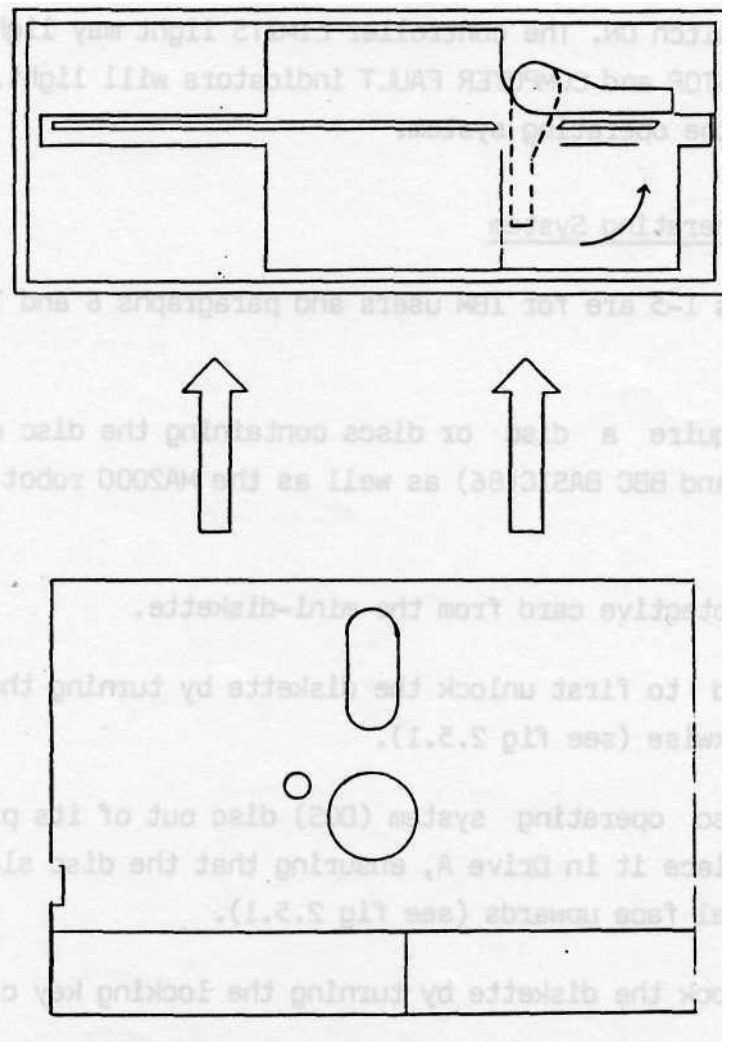


FIG 2.5.1

7. Because of limited memory space, the BBC will require you to answer the following questions:

(a)	CONTINUOUS PATH (Y/N)?	Type Y
(b)	How many SEQUENCE STEPS do you need?	Type 30
(c)	OFF LINE TEACHING (Y/N)?	Type Y
(d)	TUTOR TEXT (Y/N)?	Type Y
(e)	USER MODULE (Y/N)?	Type Y

The computer will then respond with:

```
Room for 30 steps & 149 CP u steps
       is this sufficient (Y/N)?        Type Y
```

8. Once steps 1 to 5 or steps 6 and 7 have been completed, the computer responds with:

```
                TECQUIPMENT INTERNATIONAL LTD
                    Nottingham, ENGLAND.

                          MA2000

                         Issue 3A.03

            Copyright (c)  1987  TECQUIPMENT LTD

            CLEAR DATA AREA?  (Y/N)
```

The COMPUTER FAULT light should remain lit on the controller front panel.

This light will only go off when the software is actually running.

3. SAFETY PROCEDURES AND ROBOT TESTING

The MA2000 controller can be used as a safety and test device. The front face panel of the controller should always be within easy reach of the user. On the left hand side of the panel a push-button switch labelled EMERGENCY STOP has been installed. You will also notice a similar button labelled STOP amongst the bank of switches on the right hand side of the panel. Both these switches offer a safety facility. The remaining front panel controls are Robot Test controls.

We will first discuss the 'Emergency Stop' and 'Stop' buttons.

3.1 Emergency Stop and Stop

If the robot ever behaves unexpectedly or dangerously, press either the Emergency Stop or Stop button. This will halt the robot motion immediately.

The STOP button halts the robot instantaneously, but by then pressing Auto, also on the controller panel, the robot will continue from where it left off.

The EMERGENCY STOP button also halts the robot instantaneously, but does so by cutting out the robot power supply. The robot will become dynamically unstable and will drift in position. Thus, when power is resumed it will continue from a slightly different position. This new position may be out of the robot's limits (or operating range) and certain procedures must be used to bring the robot back into its limits (see section 3.2).

The EMERGENCY STOP indicator may automatically light if the robot is moved out of its limits when driving. Again, the robot must be brought back into its limits (see section 3.2).

3.2 Robot Testing by Local Control

All individual six axes of the robot are restricted in operation. When one, or more, of the robot joints has moved to a position outside its operating range, the robot is said to be OUT OF LIMITS.

The green LIMITS light on the controller front panel only remains lit when all six axes are within their operating range, i.e. IN LIMITS.

Whenever the limits light switches OFF, the controller will assume total control over the robot and commands made by the microcomputer and keyboard will be ignored. The robot must be brought back into limits before computer/keyboard control is resumed.

The following robot TEST procedure is that also used to move the offending joint(s) back into LIMITS.

(If the robot is in LIMITS after loading the operating system, it is still advisable to test the individual joints separately before continuing to section 4).

The operating system should always be loaded before testing the robot. If this has not already been done, refer to section 2.5 and follow the loading instructions.

Having loaded the operating system, the screen will display:

CLEAR DATA AREA	type (Y) on the ucomp. keyboard

A flashing message will then say:

CONTROLLER NOT IN AUTO	press AUTO and then RESET on the controller front panel.

TEST PROCEDURE

1. Press the STOP button on the controller front panel.

This switches the controller from Automatic (AUTO) to manual (or local control) (STOP) allowing the user to test the robot.

2. Turn the selector dial on the right hand side of the controller to your desired robot joint. The abbreviations around the dial are:

WST	Waist	PCH	Pitch
SHD	Shoulder	YAW	Yaw
ELB	Elbow	ROL	Roll

Try WST to begin with. Press TEST and RESET together.

3. Press and hold the controller TEST button.

The selected joint should rotate about its axis. The INCREASE button rotates the joint clockwise, the DECREASE button anti-clockwise.

4. Try selecting the other joints with the dial and repeat the test procedure.

Having completed the test for all joints, check that the LIMITS light is lit. If the light does not show, use the test procedure to move the joints until the robot is back in LIMITS. Otherwise proceed to section 4.

4. DRIVING THE MA2000 ROBOT

IMPORTANT!

Programming and driving the MA2000 robot requires patience and skill. If you are an inexperienced user and at any time feel anxious about the robot's behaviour, remember to press the controller EMERGENCY STOP button to halt the robot (see section 3.1).

If you ever make a typing or keypad entry error, then press <ESCAPE> on the keypad and continue.

The instructions contained in the manual should directly correspond to those in the MA2000C user edit software. Do not use the manual as your only user guide. The display instructions are there to help you and should be carefully observed.

4.1 Position Programming

4.1.1 Steps and Sequences

The MA2000 Robot can be taught to perform a series of movements by programming it to follow a sequence of steps. Programming the robot involves feeding positional and control data to the robot using either the keypad or microprocessor keyboard as an interface.

The robot sequence will always originate from a position termed the 'Park Position'. This is the position that the robot will automatically move to when it is switched on (see fig 4.1.1) and is called 'Step 0'.

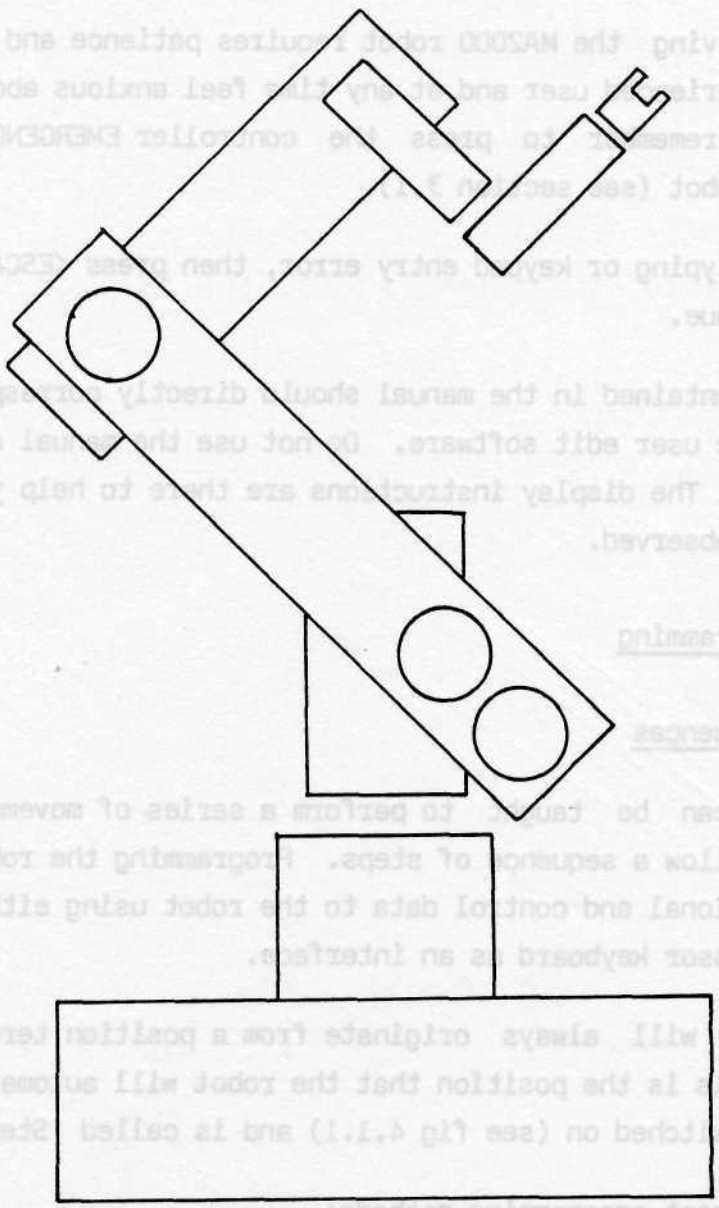
There are four different programming methods:

1. Point to point
2. Lead by the Nose
3. Continuous Path
4. Offline

'Point to Point' programming uses the keypad to move the robot through the sequence of steps. It is perhaps the easiest method to grasp and for this reason will be the first considered.

STEP 0 & STEP 1	

Fig 4.1.1



RATE	MODE	INPUT	OUTPUT	WAIT		JUMP
7	2	0	0	0		0
WAIST	SHOULDER	ELBOW	PITCH	YAW	ROLL	GRIPPER
500	400	100	500	500	500	0

4.1.2 Display Format

Each step in a sequence is defined by a bank of position and control information. The information is conveyed to the user as a grid of numbers on the display in the following manner:

```
rate      mode     input    output   wait            jump
waist  shoulder   elbow    pitch     yaw     roll  gripper
```

The upper section of the grid represents CONTROL data, the lower section POSITION data.

All the control numbers have default values. For this reason we can program positional data and drive the robot without altering any control data. An inexperienced robot driver can learn about position programming without tackling control programming at the same time. Hence the learning process is considerably simplified.

The range of numbers allocated to positional definition is:

	waist	shoulder	elbow	pitch	yaw	roll	gripper
grid	000-	000-	000-	000-	000-	000-	open = 0
range	999	999	999	999	999	999	closed = 1

The numbers represent the angle through which each joint turns to reach its destination. The angle ranges are 0° to 270° for the major axes (WAIST, SHOULDER, ELBOW) and 0° to 180° for the minor axes (PITCH, YAW, ROLL).

Fig 4.1.1 shows the robot Park Position and its corresponding grid.

4.1.3 A Trial Sequence (Point to Point Programming)

If you have already switched on the equipment, loaded the operating system, and tested the robot, ignore instructions 1 to 3.

1. Switch on the mains power supply to the equipment (see section 2.3).

2. Load the operating system with the MA2000c disc (see section 2.4).

Caution: Ensure that the limits lamp on the controller is lit before proceeding.

3. If necessary use the robot test procedure detailed in section 3.2).

The display should be flashing the message:

CONTROLLER NOT IN AUTO

4. Press AUTO and the RESET on the controller front panel.

The display should then read:

Check that it is
SAFE to MOVE the ROBOT
Then:
PRESS <EDIT> FOR TUTOR TEXT
To PARK the ROBOT press <STEP>
To MOVE the ROBOT press <DRIVE>

5. Check that there are no obstructions preventing the robot from transferring to its Park Position. The Park Position is shown in Fig 4.1.1. It may be helpful to refer to the diagram.

6. Press <EDIT> on the keypad to select the tutor text.

The display should then read:

This robot has six motions:
WAIST, SHOULDER, ELBOW,
PITCH, YAW & ROLL and a GRIPPER

STEP 0) 7 2 0 0 0 0
497 293 68 484 492 491 0

The display format used is:

STEP No) RATE MODE IN OUT WAIT JUMP
POSITION VALUES & GRIPPER

Give your instructions to the robot
using the CONTROL KEYPAD.

I will tell you which <KEYS>
to use at each stage

Press <STEP> to continue

This is a supplementary screen text provided to help inexperienced users.
The display will now inform the user of the robot motions and display format. It may be helpful to refer to section 4.1.2.

7. Ensure that the robot operating envelope is FREE from OBSTRUCTION and then press <STEP> on the keypad to continue.
The display should read:

Now
To PARK the ROBOT press <STEP>
To MOVE the ROBOT press <DRIVE>

8. Press <STEP> again and the robot should now move to its PARK POSITION. If the robot behaves dangerously or unexpectedly press the EMERGENCY STOP button and refer to section 3.1.

9. The screen should display:

MOVING TO PARK POSITION

Step 0) 7 2 0 0 0 0
500 400 100 500 500 500 0

Stored sequence number 0
Contains 0 steps & 0 CPusteps
Room for 250 steps & 1406 CPusteps

HOLDING AT STEP 0

This is the initial default step.
Press
<EDIT> to teach the robot what to do.

The lower grid numbers should already be familiar to the user. They are the PARK POSITION numbers mentioned in section 4.1.2.

Ignore the upper grid numbers for now; they are the default control data values.

Note: if in the following operations you press the wrong keys and inadvertently give control to the host computer, then just press 0 <RETURN> to get back to the keypad.

10. Press <EDIT> (to teach MA2000 what to do).
The display should now read:

```
EDIT   with tutor

STEP   0)      7   2   0   0   0   0
         500 400 100 500 500 500  0

Press:
<STEP> <STEP> to select next step.
```

11. Press <STEP> <STEP>
The display should now read:

```
EDIT   with tutor

STEP   1)      0   0   0   0   0   0
           0   0   0   0   0   0   0  0

Press:
<DRIVE> to copy the present position
                into this new step
```

12. Press <STEP> <STEP>.
This sets up the grid for the next step in the sequence, in this case STEP 1 the first step of the sequence. Whilst gaining experience it is recommended that STEP 1 be left as a park position.

13. Press <DRIVE>.
This enters the present robot position and control information into the step (1) grid. The Park Position data is therefore copied into step (1). The purpose of this is to provide the system with a reference position from which the robot will move.

14. The screen will now display:

```
DRIVE  with tutor

Step 1)      0   0   0   0   0   0
       500 400 100 500 500 500  0

Use the
<MOTION KEYS> to move the robot
                to required position

The <DRIVE> key can be used to toggle
            between SLOW and RAPID drive

Then press:

<LEARN> to store the current
               positions into the step

<ESCAPE> to keep original values
```

15. Press <LEARN> - we wish to keep step 1 as a park position.
    Press <STEP> <STEP> - move on to step 2.
    Press <DRIVE> - copy step 1 into step 2 for a reference.

    WE ARE NOW READY TO MOVE THE ROBOT AND TEACH IT A NEW POSITION.

    The journey from the Park Position at step 1 to the new position is called STEP (2).

16. Press <DRIVE> to select RAPID drive.
    Note : Rapid DRIVE TUTOR is in inverse video at the top of the screen.

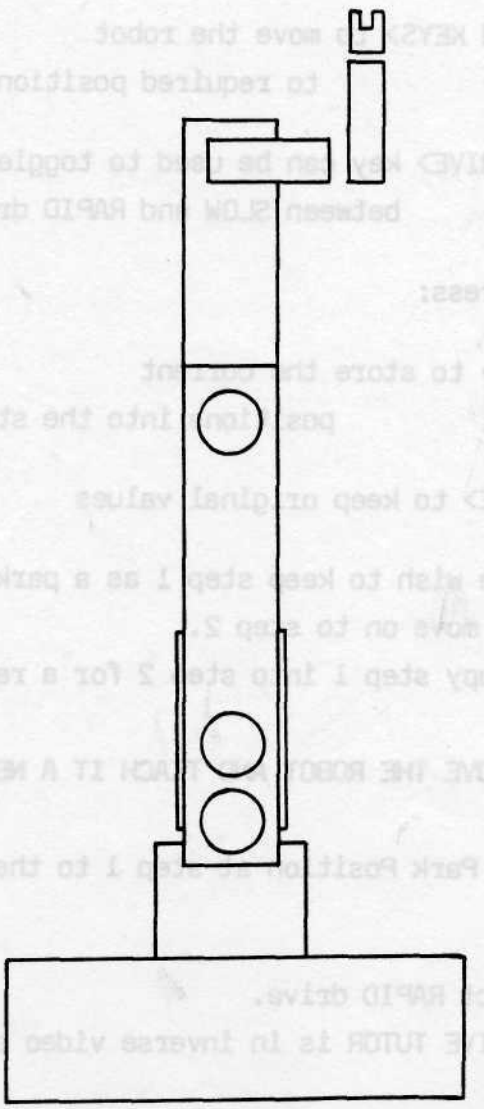
17. Press the motion key labelled <SHOULDER>.
    The robot's shoulder should move in a clockwise sense. The grid number representing the angular displacement of the shoulder should increase.

18. Press <DRIVE> to select <SLOW> drive.

19. Press the <SHOULDER> motion key again.
    The shoulder joint will now only displace very slightly for each nudge of the key.

STEP 2	

Fig 4.1.2



RATE	MODE	INPUT	OUTPUT	WAIT		JUMP
7	2	0	0	0		0
WAIST	SHOULDER	ELBOW	PITCH	YAW	ROLL	GRIPPER
500	500	500	500	500	500	0

You can toggle between rapid and slow drive by pressing the <DRIVE> key.

20. Try some of the other motion keys. Continue when familiar with the motion keys.
    The following four diagrams demonstrate a few robot positions and their coresponding positional grid values.
    Match the data values shown on the step (2) example sheet to those on your screen using the motion keys. When the values perfectly agree, your robot should be in a vertical position (see Fig 4.1.2).
    It is a good idea to move to roughly the desired position on RAPID drive and to move the last degree or so on nudge control.

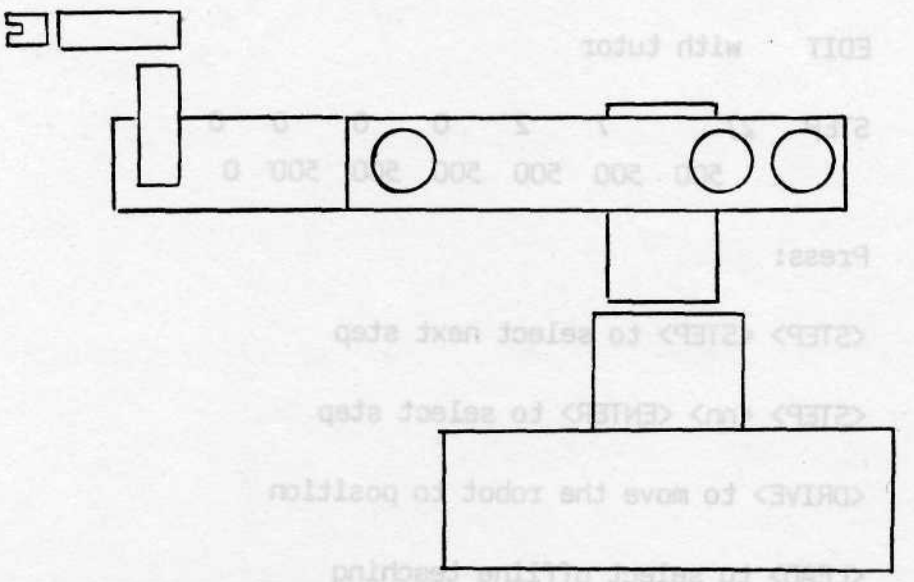
21. When you are satisfied with the position press <LEARN>.
    The computer will store this new position in step (2). Whenever the robot is asked to perform step (2) it will move to the vertical position.
    The display should nor read:

```
EDIT    with tutor

STEP   2)       7    2    0    0    0   0
         500  500  500  500  500  500  0

Press:

<STEP> <STEP> to select next step

<STEP> <nn> <ENTER> to select step

<DRIVE> to move the robot to position

<LEAD> to select offline teaching

<RUN> to execute the taught steps

<ENTER> for further instructions
```

Note: Version 4A users will have an additional line:

```
<PATH> to toggle XYZ co-ordinates.
```

Fig 4.1.3



RATE	MODE	INPUT	OUTPUT	WAIT		JUMP
7	2	0	0	0		0
WAIST	SHOULDER	ELBOW	PITCH	YAW	ROLL	GRIPPER
500	135	500	500	500	500	0

22. Press <STEP> <STEP>.
    The information grid for step (3) should now be displayed on the monitor.

23. Press <DRIVE> to copy the present robot position into step (3) as a reference position.

24. Use the motion keys to position the robot horizontally (see Fig 4.1.3).

25. When you are satisfied with the position press <LEARN>.
    The computer will store this new position in step (3) and the robot will move to the horizontal position whenever it is asked to perform step (3).
    The positional programming procedure is therefore:

<STEP> <STEP>	to select next step grid in sequence
<DRIVE	to enter reference position
<MOTION>	to move robot to new destination
<LEARN>	to store the positional data in step that has just been completed.

26. Using the above procedure, teach the robot a fourth step by matching the positional values to those on example sheet step 4 (see Fig 4.1.4).
    Don't forget to press <LEARN> to store the position.

Having completed the short sequence of steps, we now wish to run through the steps.

4.1.4 Running a Sequence

(a) Press <STEP> <1> <ENTER>

(b) Press <DRIVE>. The robot should move to the Park Position.

(c) Press <LEARN>

(d) Press <RUN> <RUN>. This executes the entire sequence.

The robot should now run through the positions and stop when it has reached step (4).

The display will now read:

```
Press <HOLD> to halt execution

STEP   2)      7    2    0    0    0   0
         500  500  500  500  500  500  0

STEP   3)      7    2    0    0    0   0
         500  135  500  500  500  500  0

STEP   4)      7    2    0    0    0   0
         500  135  140  500  500  500  0

STEP   5)      0    0    0    0    0   0
            0    0    0    0    0    0  0

EXECUTION FAULT at STEP 5

Sequence    0 halted
```

Ignore the comment "EXECUTION FAULT at STEP 5". It simply tells us that step 5 is an empty step and cannot therefore be executed as part of the sequence.

Whenever the comment does appear, run through the following robot Park Procedure before continuing.

Robot Park Procedure after an Execution Fault

(a) Press <STEP> to continue.

(b) Press <EDIT>.

(c) Press <STEP> <1> <ENTER>.

(d) Press <DRIVE>. (Robot should move to Park Position with this command).

(e) Press <LEARN>.

Stepping a Sequence

Once the robot has been returned to step 1 the sequence can be stepped through as follows:

(a) Press <RUN>

(b) Press <STEP> to observe each step consecutively.
(c) On reaching step 4 press <EDIT> to take you back to tutor text.

Now try adding a few additional steps of your own to the sequence.

If necessary, refer to section 4.1.3 and create 3 new steps. Call your steps step 5, step 6 and step 7. Then refer to section 4.1.4 to run the sequence.

An 'execution fault' message will still appear on the screen upon completion of the program RUN. This time however, the execution fault will be found at step 8; this is the first empty step reached.

When you feel confident enough to further your 'robot education'

Press <EDIT>
Press <CLEAR> <5> <ENTER> <7> <ENTER>

This will empty the data grids for the additional steps that you have created and leave the original 3 steps unaffected for use in the next section.

WE ARE NOW READY TO LEARN ABOUT
ROBOT CONTROL PROGRAMMING.

4.2 Control Programming

4.2.1 Programming the Data Grid

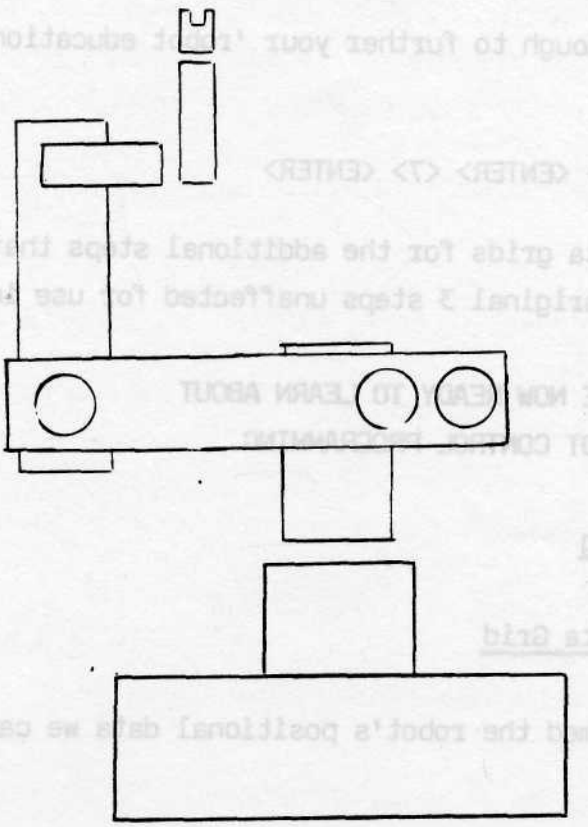
Now that we have programmed the robot's positional data we can edit the control parameters.

The control data grid is the upper section of the information grid and has the following format (also see Fig 4.2.1).

RATE	MODE	INPUT	OUTPUT	WAIT	JUMP	control grid (with default values)
7	2	0	0	0	0	

The numbers in the above table are the control default values. These values will be automatically assumed unless the user makes any alteration to them.

Fig 4.1.4



RATE	MODE	INPUT	OUTPUT	WAIT		JUMP
7	2	0	0	0		0
WAIST	SHOULDER	ELBOW	PITCH	YAW	ROLL	GRIPPER
500	135	140	500	500	500	0

The following table shows the range of numbers allocated to each cell in the control grid.

RATE	MODE	INPUT	OUTPUT	WAIT	JUMP	control grid (with cell ranges)
1-9	1-9	1-4 (-1)-(-4)	1 or 4 (-1)or(-4)	0-255	0-STEP X '0' means jump function inhibited	

We will consider the rate, wait and jump control parameters.

4.2.2 Rate Command

Rate simply determines the speed at which the robot will travel when it performs the step. A value of Rate = 1 is the slowest speed possible. The maximum travelling speed is achieved when Rate is set to 9.

We will now alter the rate for step (4) in the example sequence considered in section 4.1.1.

If the screen shows: 'EXECUTION FAULT AT STEP X'

Then run through the Park procedure on page 34. Otherwise continue.

1. Press <STEP> <4> <ENTER>
   The step (4) information grid should now show on the monitor screen.

2. Press <RATE> <4> <ENTER> RATE COMMAND
   This sets the step 4 rate cell to rate '4'. Note the cell number change in the control grid (see Fig 4.2.2).
   Now run the sequence.

3. Press <RUN> <RUN>.
   The final step in the sequence, step (4), will move considerably slower than the preceding steps.
   The screen will display: 'EXECUTION FAULT AT STEP 5'. Use the procedure on page 34 to park the robot before continuing.

4.2.3 Wait Command

The computer will usually send the grid information to the robot as soon as the previous one has been completed. If you require the robot to

wait at the end of a given step before proceeding to the next, use the WAIT command.

The control data grid contains a cell labelled WAIT (see Table 1).

The number range for the cell is 0-255, each unit representing a wait for 1 second. The maximum wait is therefore 255 seconds.

We will now edit the step 3 control grid in the example sequence to create a wait of 8 seconds between step 3 and step 4.

1. Press <STEP> <3> <ENTER>
   The step (3) information grid should now show on the monitor screen.

2. Press <WAIT> <8> <ENTER> WAIT COMMAND
   This sets the step (3) wait cell to wait '8'. Note the cell number change in the control grid (see Fig 4.1.3).
   Now run the sequence.

3. Press <RUN> <RUN>.
   The robot will wait 8 seconds before commencing step (4) in the example sequence. Step (4) will still move at a slower rate.

   The screen will display: 'EXECUTION FAULT AT STEP 5'. Use the park procedure in section 4.1.4 to park the robot before continuing.

4.2.4 Jump Command

The JUMP command is an extremely useful feature. It can be used to either cycle the sequence around or to insert additional step (5) between existing adjacent steps.

The control grid contains a cell labelled JUMP. The number range for the cell is '-' to 'X'. (Where X is the number of the step we are JUMPING to).

In our example sequence we have 4 steps. If we wish to cycle continuously around the sequence we need to alter the JUMP cell in the step (4) control grid. The cell currently has a default value of '0'. Therefore each time step (4) is completed the program JUMPS to step (5). Here a control grid containing no data whatsoever is found. The sequence is

broken and the message 'execution fault at step (4)' is displayed. Disrupting the program flow in this way has allowed us to run our sequence through a single time only. This has minimised confusion in other sections and simplified the learning process.

The flow sequence for the existing JUMP condition is:

PARK POSITION STEP (1)

start

STEP (2)

EXAMPLE SEQUENCE

STEP (3)

STEP (4)

STEP (5) (EMPTY STEP) - 'EXECUTION FAULT'

If we now change the jump cell in the step (4) control grid from '0' to '2' the sequence will JUMP to step (2) upon each completion of step (4).

The flow sequence will then be:

PARK POSITION STEP (1)

start

STEP (2) VERTICAL POSITION

EXAMPLE SEQUENCE

STEP (3) HORIZONTAL POSITION

STEP (4) 'RIGHT ANGLE' POSITION

JUMP TO STEP (2)

1. Press <STEP> <4> <ENTER>
2. Press <JUMP> <2> <ENTER> JUMP COMMAND
   Now run the sequence.
3. Press <RUN> <RUN>.

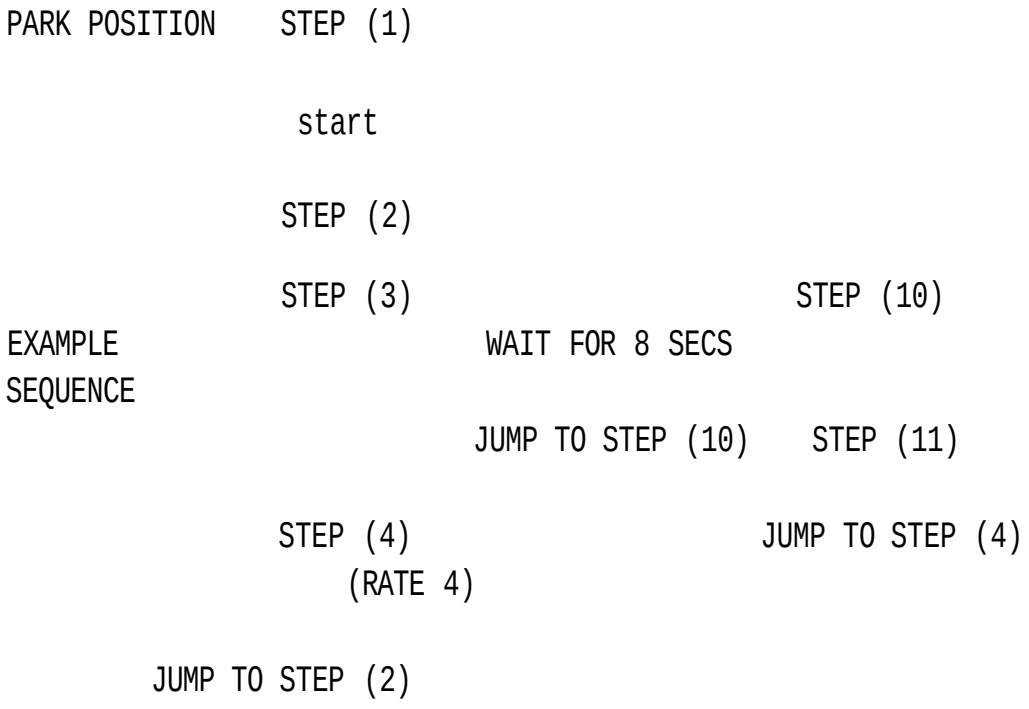
The program will now cycle around the sequence of steps continuously. Note that the Park Position is not included in the cycle.

Press <HOLD> when you wish to halt the cycle. The cycle should be broken at the end of the step in which you pressed <HOLD>.

Imagine now that we had written a sequence and accidentally omitted one or more of the steps. If the sequence was of any considerable length, correcting such an error would be very inconvenient without use of the <JUMP> command. The command allows the user to insert an additional step(s) between two existing steps in a sequence.

We will insert two new steps between step (3) and step (4) in the example sequence. We will call the new steps step (10) and step (11).

The flow sequence will then be:



The program will reach step (3) and upon completion of that step transfer to step (10). The program jumps back to step (4) of the example sequence upon completion of step (11).

1. Press <EDIT> to return to edit.

2. Press <STEP> <3> <ENTER>

   The step (3) information grid should now show on the monitor screen.

3. Press <JUMP> <10> <ENTER> (JUMP TO SUPPLEMENTARY SEQUENCE).

4. Press <STEP> <10> <ENTER>.

   The information grid for step (10) should now show on the monitor screen.

5. Press <DRIVE>.

   NOW USE THE MOTION KEYS ON THE KEYPAD TO CREATE A NEW ROBOT POSITION.

6. Press <LEARN> when you are satisfied with the new position.

7. Press <STEP> <STEP> to display the following step data grid, i.e. that for step (11).

8. Press <DRIVE>.

   Again use the motion keys on the control keypad to create a new robot position.

9. Press <LEARN> when you are satisfied with the new position.

10. Press <JUMP> <4> <ENTER> (Jump back to step (4) of the example sequence).

11. Press <STEP> <1> <ENTER>

12. Press <DRIVE> Returns robot to Park Position.

13. Press <LEARN>

    Now run the program.

14. Press <RUN> <RUN>.

The robot will then: Perform steps (2) and (3)

JUMP to step (10)

Perform steps (10) and (11)

JUMP to step (4)

Perform step (4)

JUMP to step (2)

4.2.5 The Demonstration Sequence

Note: In the following tables <ENTER> will be shown as <e>.

1. The first thing we have to do is: stop the robot executing the current sequence. Follow the procedure in table 1.

TABLE 1

KEYPAD SEQUENCE (1)	COMMENT (2)
<HOLD>	Halts the current sequence.
<EDIT>	Returns to edit.
<STEP> <1> <e>	Fetches step 1.
<DRIVE>	Gets robot to GO to step 1.
<EDIT>	Returns to edit.
<CLEAR> <1> <e> <4> <e>	Clears current program.
<EDIT>	
<PR0G>	Display asks for sequence No.
<-> <1> <e>	Demonstration sequence is transferred. See section 11 for listings.

2. The demonstration sequence has now been down loaded. Tutor text can be toggled ON/OFF by successive key strokes of <EDIT>. Run the first part of the sequence by carrying out the instructions detailed in table 2.

TABLE 2

KEYPAD SEQUENCE (1)	COMMENT (2)
<STEP>	Robot holds at park position. Step 1 is displayed.
<STEP> <STEP>	Step 2 is displayed.
<STEP> <STEP>	Step 3 is displayed. Note the 7 second delay.
<STEP> <STEP>	Step 4 is displayed.
<STEP> <STEP>	Step 5 is displayed. Note the jump to step 1.
<RUN>	Message displayed: "Sequence -1 starting at STEP 1"
<RUN>	Message displayed: "Press <HOLD> to halt execution". Robot commences sequence.

The next part of the demonstration introduces INPUT, OUTPUT, WAIT and JUMP and their use in MODE 1. The robot control unit has four input connections and four output pairs (see Fig 6.1). These are for use when the robot works in conjunction with other automated tools. You will now need the decision box connected at para 2.4. To proceed, carry out the instructions detailed in table 3.

TABLE 3

KEYPAD SEQUENCE (1)	COMMENT (2)
<HOLD>	Stops the current sequence.
<EDIT> <STEP>	Displays step 5 for editing.
<JUMP>	Cancel jump to step 1.
<RUN>	Allows entry to this part of the sequence
<STEP>	Allows robot to move through each step, holding each time.
Press <STEP> as required:	until "HOLDING AT STEP 6" and step 6 parameters are displayed. Note: output 3 is set.
<EDIT>	To get back to edit.
<STEP> <STEP>	Step 7 is displayed. Note: output 3 has been cancelled (-3).
<STEP> <STEP>	Step 8 displayed. Note that only commands are displayed. This is MODE 1, a "conditional jump". If input 3 on the controller is not set within 5 seconds (WAIT 5) then the next step is step 6, otherwise step 9. The decision box can be used to provide input 3.
<STEP> <STEP>	Displays step 9.
<STEP> <STEP>	Displays step 10. Note the jump to step 6.
<RUN> <RUN>	Starts the sequence running.

4. This part of the demonstration sequence introduces MODE 3, in which position values of 500 are used to determine the sign of the rel ative step. For example a value of 525 means add 25 to the value of the previous limb position whilst 460 means subtract 40;
i.e.
new limb position = previous limb position + (position value - 500).

Note: 500 is the mid range position and as the position values are stored as two unsigned bytes in the computer. Therefore when teaching the robot in MODE 3, the differences in limbs position value between successive steps must be less than 500.

In step 15 of this sequence the JUMP is set to -1: it is the signal to return from a subsequence. It points the program to the line after the line from which the jumps occured. (See Fig 4.2.1).

To proceed carry out the instructions detailed in table 4.

TABLE 4

KEYPAD SEQUENCE (1)	COMMENT (2)
<HOLD>	Stop the current sequence.
<EDIT> <STEP> <8> <e>	Displays step 8 for editing.
<JUMP> <0> <e>	Remove the jump to step 6.
<STEP> <10> <e>	Displays step 10 for editing.
<JUMP> <0> <e>	Removes the jump to step 6.
<RUN>	Allows entry to this part of the sequence.
<STEP>	Moves robot to next step of sequence.
Press <STEP> as many times as required to reach step 11.	
<EDIT>	To get back to edit. Note: jump to 14 of step 11.
<STEP> <STEP>	To display step 12. Note: jump to 14.
<STEP> <STEP>	To display step 13. Note: jump to 11.
<STEP> <STEP>	To display step 14. Note: Mode 3.
<STEP> <STEP>	To display step 15. Note: Mode 3 and jump -1. (Return from subsequence).
<RUN> <RUN>	Start sequence running.

Introducing MODE 4

5. In this mode the robot moves as in mode 2, from its point of origin to its destination, searching for a specified INPUT which may be expected somewhere along its path. When found, the search path is abandoned and the next step executed. If the specified INPUT is not generated then the robot will execute the next step on reaching its destination. To proceed, carry out the instructions detailed in table 5.

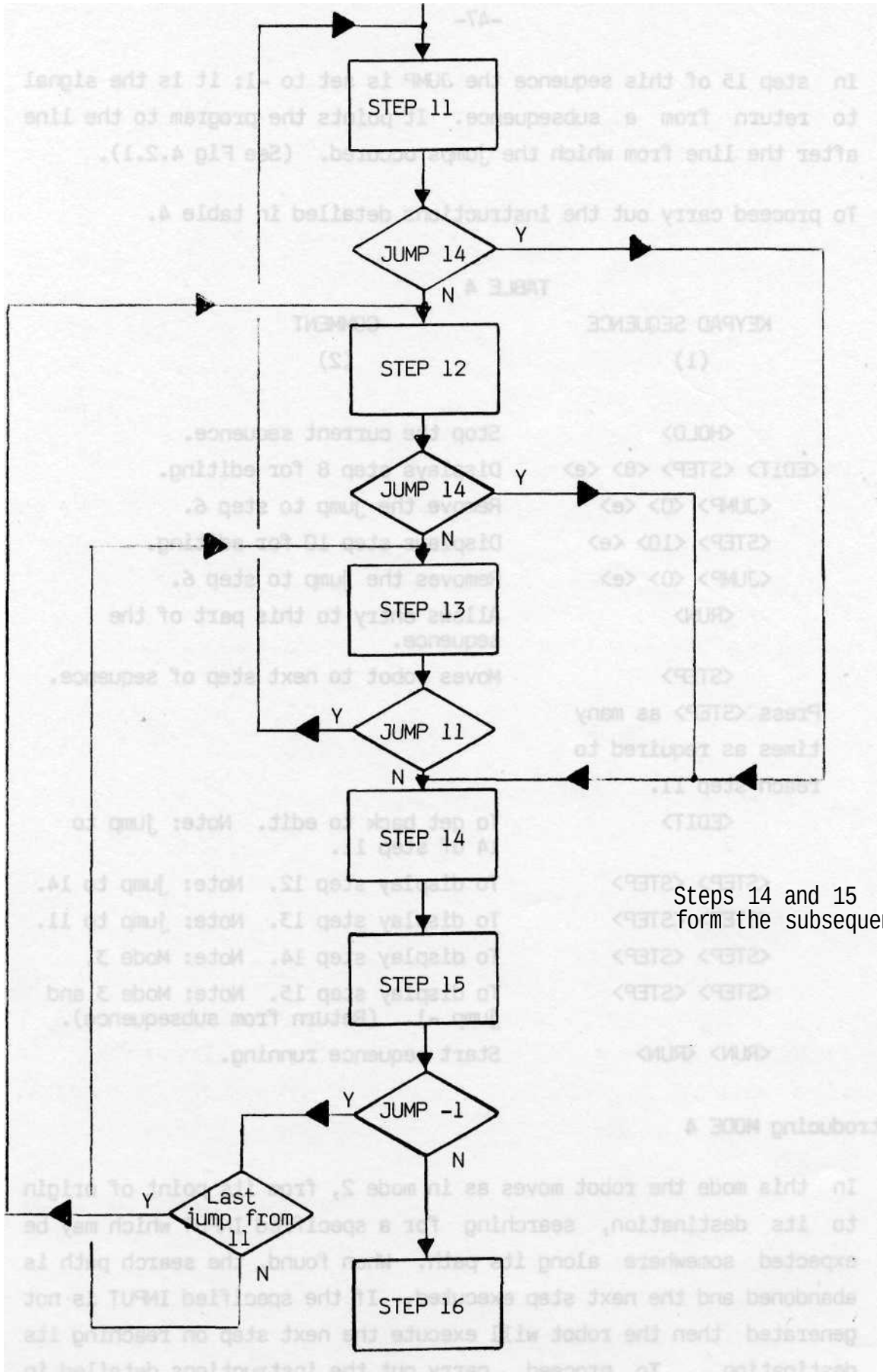


Fig 4.2.1 Demonstration Sequence Step 11 to Step 15

TABLE 5

KEYPAD SEQUENCE (1)	COMMENT (2)
<HOLD>	Stop the current sequence.
<EDIT> <STEP> <1> <3> <e>	Displays step 13 for editing.
<JUMP> <16> <e>	Remove the jump to step 11 and replace it with a jump to step 16.
<RUN>	Allows entry to the sequence.
<STEP>	Moves robot to next step of sequence.
Press <STEP> as many times as required to reach step 16.	
<EDIT>	To get back to edit. Step 16 displayed.
<STEP> <STEP>	Step 17 displayed. Note: Mode 1 with input 1 set. This is the indefinite wait. The robot holds this step until input 1 is present.
<STEP> <STEP>	Step 18 displayed. Note: this is MODE 4 and has INPUT 2 set.
<STEP> <STEP>	Step 19 displayed. Note: this is MODE 4 and has INPUT 3 set as well as JUMP.
<RUN> <RUN>	Starts sequence running. Use the decision box as appropriate and note what happens when the inputs are not forthcoming.

Introducing MODE 5

6. In this mode the robot behaves as in mode 4, but also stores the position at which a particular input was sensed. These self taught positions are then available to be used as servo motion commands in later steps of the sequence. To proceed, carry out the instructions detailed in table 6.

7. When you are satisfied that you understand the operations, try running the complete sequence from step 1.

TABLE 6

KEYPAD SEQUENCE (1)	COMMENT (2)
<HOLD>	Stop the current sequence.
<EDIT> <STEP>	Displays step 19 for editing.
<JUMP>	Remove the jump to step 16.
<RUN>	Allows entry to the sequence.
<STEP>	Moves robot to next step of sequence.
Press <STEP> as many times as required to reach step 20.	
<EDIT>	To get back to edit. Step 20 displayed.
<STEP> <STEP>	Step 21 displayed. Note: Mode 5, input 4 set and JUMP 26.
<STEP> <STEP>	Step 22 displayed, MODE 5. Note: INPUT and JUMP.
<STEP> <STEP>	Step 23 displayed, MODE 5. Note: INPUT and JUMP.
<STEP> <STEP>	Step 24 displayed, MODE 5. Note: INPUT and JUMP.
<STEP> <STEP>	Step 25 displayed MODE 2.
<STEP> <STEP>	Step 26 displayed MODE 2. Note the zero values for the servo motions.
<STEP> <STEP>	Step 27 displayed. Note zero values.
<STEP> <STEP>	Step 28 displayed. Note zero values.
<STEP> <STEP>	Step 29 displayed. Note zero values.
<STEP> <STEP>	Step 30.
<RUN>	Allows entry to the sequence, which we will run one step at a time.
<STEP>	Moves robot to next step in sequence. Press input 4 on the decision box at arbitrary point during movement.
<STEP>	As above, press input 3 on decision box at an arbitrary point during movement.
<STEP>	As above, press input 1.
<STEP>	As above, press input 2.
<EDIT>	To get back to editor.
<LIST><2><l><e><2><9><e>	List steps 21 to 29. Note: steps 26 to 29 now have data in them which relates to the position at which the input was set.
<EDIT>	
<STEP><2><0><e>	Edit step 20.
<JUMP><2><5><e>	Steps 21 to 24 have served their purpose, therefore jump to 25.
<RUN><RUN>	Runs the sequence.

4.3 Additional Position Programming Techniques

Recall that the previous section (4.1.1) used a technique known as 'Point to Point' programming to teach the robot its position information. The motion keys on the keypad were used to alter the robot's position. When the desired position was attained the position data was fed to a display grid and consequently taught to the robot.

The following three procedures offer the user other programming methods. You will find each method suitable to a particular robot application.

Halt the previous sequence if you have not already done so.

Press <EDIT> to return to edit.
Press <STEP> <1> <e>.
Press <DRIVE> to get the robot to drive to step 1.
Press <shoulder> drive key until shoulder position indicates 400.
Press <elbow> drive key until elbow position indicates 100.
Press <LEARN> to store this step.
Press <EDIT> to return to edit.

Clear the previous sequences as follows:

Press <CLEAR> <2> <e> <3> <0> <e> clearing steps 2 to 30.
Step 1 is to be retained as our park position.
Press <EDIT> to return to edit.

4.3.1 Lead-by-the-Nose

Lead by the nose is used to manually move the robot to a desired position. The robot chooses its own path to the desired position when the sequence is executed.

We will now teach the robot the same steps as in section 4.1.1 using this different programming technique.

PROGRAMMING PROCEDURE

1. Press <STEP> <2> <ENTER> to display the step (2) information grid.

2. Press <DRIVE>. The robot is then ready for motion commands.

This time however, we will not use the motion controls on the key-pad. Instead we will manually move the robot to our desired position - the 'vertical position'.

CAUTION: When <LEAD> is pressed the computer will bleep for about 5 seconds and the the robot's dynamic braking will be released. Ensure that the robot is supported.

Note for BBC users: the computer does not bleep and must be support-ed prior to pressing <LEAD>.

3. Press <LEAD>, wait for the computer to start bleeping and then (and only then) SUPPORT THE ROBOT IN YOUR HANDS.

4. Lead the robot slowly and gently to the vertical position to match the display values of Fig 4.1.2.

5. Press <HOLD> to restore dynamic braking.

6. Press <LEARN> to store the position.

7. To move to the next step press <STEP> <STEP>.

8. Repeat sub paragraphs 3 to 7 for steps 3 and 4, which should be aligned with Figs 4.1.3 and 4.1.4 respectively.

9. To get back to edit with tutor press the <EDIT> key after the <LEARN> key has been pressed to store the last posture position.

10. Run the sequence and think about what has to be done to make the sequence repetitive.

You will notice that the robot moves from position to position, but the path it takes is not necessarily that used in the programming stage. The computer records the desired end point and chooses the robot's path to that end point when the step is being executed.

If the user requires to teach the robot the actual path it is to take in reaching the desired end point, then Continuous Path Programming is used.

4.3.2 Continuous Path Programming

The sequences you have created up to now have caused the MA2000 to move to distinct positions described by the 'posture' numbers ('positional grid' numbers) at each step (e.g the PARK position is 500 350 100 500 500 500).

When you run your program the robot moves between these positions in an efficient way, choosing the actual path itself so as to minimise the time between positions.

In some cases it might be necessary for the robot to follow a specific path between the positions, for instance if it is welding, paint spraying or even writing. The PATH key on the key pad accesses this facility to the user.

To teach the MA2000 a CONTINUOUS PATH it is necessary to have the host computer memory mapped out differently from when the continuous path is not being used.

Halt the sequence if you have not already done so. Return to the tutor text and get the robot to move to the park position (step 1).

PROGRAMMING PROCEDURE

1. Press <CLEAR> <2> <ENTER> <4> <ENTER> to ensure that steps 2 to 4 information grids are empty.

2. Press <STEP> <2> <ENTER> to display the step (2) data grid.

3. Press <DRIVE>. The robot is then ready for motion commands.

   Again we will manually move the robot arm.

   In any sequence there can be only 1 step which is continuous path. With the standard software loaded (MA2000 disc) there is enough memory in the IBM microcomputer for at least 50 or 60 seconds of Continuous Path teaching, but if you teach for less than this time, you can still only have 1 step containing continuous information.

CAUTION; When <PATH> is pressed the computer will bleep for about 5 seconds then the robot's dynamic braking will be released. Ensure that the robot is supported.

Note for BBC users: Unless you have answered 'yes' to the question "CONTINUOUS PATH (Y/N)?" on initialisation then you will not be able to do this. Once <PATH> is pressed, dynamic braking is released immediately and the computer does not give any warning bleeps. SUPPORT THE ROBOT IN YOUR HANDS NOW.

4. Press <PATH>, wait for the computer to start bleeping and then (and only then) SUPPORT THE ROBOT IN YOUR HANDS.

5. Now gently and slowly lead the robot through the chosen path taking care not to move the arm out of limits.

   Note: The continuous path is stored in the computer memory as a series of u steps. When the memory is full the computer will bleep and dynamic breaking will be restored. It is quite possible that your arm will tire before computer memory space is exhausted.

6. Press <HOLD> to restore dynamic braking.

7. Press <LEARN> to store the steps.

8. Press <EDIT> to get back to edit.

9. Press <STEP> <3> <ENTER> to display the step (3) information grid.

10. Press <DRIVE>

    The robot is now ready for drive motion commands.

11. Use the motion keys on the keypad to create a position of your own choice.

12. Press <LEARN>

13. Press <JUMP> <2> <ENTER> JUMP COMMAND to cycle around the sequence.

14. Park the robot.

    Press <RUN> <RUN> to execute the program.

The robot will follow the path you taught it in step (2) (the Continuous Path) and continue to the step (3) position. The sequence will cycle around until you press <HOLD>.

If you accidently move the robot arm OUT OF LIMITS when Continuous Path Programming, use the Robot Test Procedure (see section 3) to bring the robot back into LIMITS and sort the Continuous Path program again.

You will notice that the RATE cell for the continuous path control grid will automatically become 5. This is the RATE DEFAULT value when operating in the Continuous Path mode.

A rate value of 6 or 7 gives an execution speed of approximately that used during programming. Rate 9 will take about half the teaching time, and RATE 1 would last up to 30 minutes.

4.3.3 Offline Programming

Offline programming uses the host computer's keyboard to set all the cells in the position and control grids to whatever value is required.

Halt the sequence by pressing <HOLD>.

PROGRAMMING PROCEDURE

1. Press <EDIT> on the keypad. Ensure that this is the edit with tutor mode.

2. Press <LEAD> on the keypad, the display now reads:

   OFFLINE INPUT of SEQUENCE STEPS

   Type
   M for Robot to MOVE to the Positions.

   Note: For version 4A (XYZ) the display prompts will first ask whether programming is to be in posture values or XYZ co-ordinates. Press <P> for posture values and the above prompt will be displayed.

3. Press <M> on the host computer keyboard as we wish the robot to move. The display now reads:

```
OFFLINE INPUT of SEQUENCE STEPS
                with Robot MOVING

Stored sequence number   0
Contains   4 steps  &    0 CPusteps
Room for 250 steps  &  1406 CPusteps

To return to EDIT type a ZERO STEP No.

Type STEP No.
```

4. Type in <4> for step number 4 followed by <RETURN>.

5. The control grid is now displayed. Type in new values of RATE, MODE, INPUT, OUTPUT, WAIT and JUMP, pressing <RETURN> after each value is entered. By just pressing <RETURN> the previous values will be entered.

6. The posture grid will now be displayed. Type in the posture values for WAIST, SHLDR, ELBOW, PITCH, YAW, ROLL and GRIP; pressing <RETURN> after each value is entered.

7. You will then be asked for the next step number; in this way a complete sequence can be taught 'offline'.

   The sequence steps can be input in any order, not necessarily in order of ascending step number. You can input the same step number several times if you change your mind or make a mistake. You can overwrite steps in an existing sequence, or add steps, or fill in the gaps, etc. The advantage of programming this way is that the robot itself is not required to move. An entire sequence can be input using the microcomputer alone.

   If you wish to get back to the keypad, then when asked for the next step number type in 0 followed by RETURN.

5. CONTROL KEYPAD

We have made considerable use of the control keypad, perhaps without realising the significance of individual keys. What follows, details the function of these keys.

5.1 Main Control Keys

<RUN> The sequence steps are executed continuously, subject to process inputs and wait instructions contained within the steps.

<HOLD> Sequence halts at END of the current step.

<STEP> Operational when on HOLD, enables the user to execute the sequence step by step.

<EDIT> Operational when on HOLD, causes entry to the sequence EDIT-ing functions which are selected using the edit-function keys.

5.2 Multi-Function Keys

The multi-function keys yield their different meanings in a logical manner:

<top left> The EDIT function as marked in the top left hand corner.

<top right>The NUMERIC value, following the selection of an EDIT function. Numeric entry is terminated by ENTER, shown as <e> in the following description.

<centre> The MOTION DRIVE functions following a keystroke to <DRIVE>.

<STEP> As well as its use under HOLD, STEP is used in the EDIT functions to select a new step,
e.g. <STEP> <7> <e> to select step 7
or: <STEP> <STEP> to select the following step
(i.e. Single Step).

<PR0G> Sets up file number for the taught sequence.
e.g. <PROG> <nn> <e>. The file number can be any positive number which is terminated by the normal enter key. Negative numbers are reserved for special system use.

<LOAD> Instructs the computer to search for and load the specified sequence file from the casette.

<SAVE> Instructs the computer to dump the sequence file to the cassette. LOAD & SAVE are only accessible immediately following the sequence numbering function,
e.g. <PROG> <nn> <e> <LOAD>

<COPY> Enables a block of steps to be copied into another block:
e.g. <COPY> <4> <e> <8> <e> copies steps 4 to 8 inclusive <2> <0> <e> <2> <4> <e> into steps 20 to 24.

<FUNCTION> This key is used to access special system utility functions, some of which should only be used under guidance (section 5.4).

<LIST> Lists sequence steps, e.g. <LIST> <5> <e> <7> <e> will list steps 5 through to 7 inclusive. To list one step, e.g. 5, then <LIST> <5> <e> <5> <e>.

<CLEAR> Clears sequence steps, i.e. set all items to 0.
e.g. <CLEAR> <1> <2> <e> <1> <5> <e> will clear steps 12 through to 15 inclusive. The steps are also listed after being cleared.

<RATE> Set up entered value in the appropriate item in the sequence step.
<MODE> e.g. <MODE> <1> <e>

<OUT> or <OUT> <-> <2> <e>
<WAIT>
<JUMP> or <JUMP> <2> <9> <e>

<ESCAPE> Any of the functions involving entry of numeric values can be terminated before taking effect by pressing the <ESCAPE> key. This can also be used to escape from a SAVE/LOAD request, or from an unintentional function selection.

<RUN> Causes exit from EDIT in HOLD mode.

<DRIVE> Only operational when in EDIT. Causes entry to the DRIVE function.

5.3 Drive Function Keys

Whenever <DRIVE> is selected during an EDIT, the robot will move to take up the position described by the STEP currently being editted. Once the robot has finished moving, the DRIVE motion keys can be used to drive the robot to a new position.

Two keys are used to control each motion, one to increase the position value, one to decrease it, as shown.

The pneumatic gripper is toggled ON/OFF by pressing the <GRIP> key.

SLOW DRIVE - this is automatically selected on entry to DRIVE.

Successive keystrokes to the same key (or holding the key down) will gradually increase, or decrease, the position of the motion.

RAPID DRIVE - pressing <DRIVE> a second time switches from slow drive to rapid drive. Only one motion at a time may be moved during EDIT, and the movement stops when the motion key is released.

Further pressing of <DRIVE> will toggle between SLOW & RAPID drive.

To exit from DRIVE: press either:

<LEARN> To save current arm positions. (This is only operational when in Drive mode, see above.)

<ESCAPE> To retain previously learnt arm positions, i.e. to cancel all the DRIVE motions which have been imposed since the beginning of the edit of this current STEP.

5.4 Utility Functions

The <FUNCTION> key on the keypad enables certain utility functions to be called up by the user. These are accessed with EDIT by pressing <FUNCTION> and then a <NUMBER> followed by <ENTER>. Some functions also request a time period to be entered for which they are active. This is entered in the usual way, e.g. <2> <0> <e> for 20 seconds. The others are exited by pressing <ESCAPE>.

<FUNCTI0N> number

<1> View the current status of the process inputs for test purposes.

<2> Enable the process outputs to be set for test purposes, e.g. pressing <2> <e> <-> <2> <e> in response to SET process output would switch on output 2 and then switch it off again.

<9><0> Report the keypad number as each key is pressed.

<9><1> Report the positional errors.

<9><2> Report the actual positions in ADC bits, where the range 500-3500 represents the span angle of 270° for major axes, and 180° for minor axes.
Note: The MA2000 resolves to 3000 bits but only controls to 1000.

<9><3> Report the motor powers.

<9><9> Report the current P.I.D. terms in the feedback loops of each of the motions, and allow them to be changed in the range 1-19. It is recommended that these terms be altered by no more than 2 units at any session, so that the effect of the change on the robot's response can be monitored. Very violent movements can be caused by inexperienced use of this feature and an "access code" is required before alterations can be made.

Note: Functions 90-91 first ask for a time in seconds for which the reporting should last.

6. DESCRIPTION OF AN MA2000 SEQUENCE

Following the introduction to robot sequences in Section 4 (where the DEMONSTRATION SEQUENCE was run), this section describes sequences in more detail.

An MA2000 sequence consists of a number of steps. Each step contains 12 items comprising 6 control commands and 6 position commands. These are displayed on the screen in the following format:

	RATE	MODE	IN	OUT	WAIT	JUMP
waist	shoulder	elbow	pitch	yaw	roll	gripper

The six motion positions have the range 0 to 999. A pneumatic gripper is fitted which has position values of 0 and 1.

RATE OF MOVEMENT

The RATE sets the speed at which the MA2000 moves between the positions contained in the sequence steps. RATE can have values 1 to 9.

For fine control, set the RATE to 1.

MODE OF MOVEMENT

Nine different modes of control are available on the MA2000.

MODE 1 executes interface INPUT/OUTPUT and WAIT and JUMP commands. In MODE 1 no move commands are sent to the robot.

If no INPUT channel is specified within a MODE 1 step, then any other MODE 1 commands within the step will be executed unconditionally. See below for descriptions of OUTPUT, WAIT and JUMP.

If an INPUT channel is specified within a MODE 1 step, then the other MODE 1 commands within the step will be acted upon together as a complex conditional command. The sequence will WAIT a specified number of seconds whilst the status of the named INPUT is scanned. If the channel is SET (i.e. is ON) at

the beginning of the WAIT period or becomes SET during the WAIT period, then any specified OUTPUT will be immediately switched and the next step of the sequence will be obeyed. Otherwise, on expiry of the WAIT period, the sequence will JUMP to the step number specified by the JUMP command.

If an INPUT channel is specified by a negative number, then the effect of the INPUT status will be the logical inverse of that described above. In this case, if the channel is NOT SET (i.e. is OFF) at the beginning of the WAIT period or becomes NOT SET during the WAIT period, then any specified OUTPUT will be immediately switched and the next step of the sequence will be obeyed. Otherwise, on expiry of the WAIT period, the sequence will JUMP to the step number specified by the JUMP command.

MODE 2 moves through INTERMEDIATE steps with speed as set by RATE, until it reaches the positions contained in the step.

MODE 3 is as for 2, but moves are RELATIVE to the current position rather than to an absolute position. This is the "MOTION RELATIVE" mode.

MODE 4 is as for 2, but SEARCHING for a specified INPUT; when found, the search path for the step is abandoned.

MODE 5 is as for 4, but also STORES the position of the motions at the moment the specified input was set. These self-taught positions are then available to be used as servo-motion commands in later steps of the sequence.

MODE 6 is reserved for CONTINUOUS PATH operations. MODE 6 cannot be selected via the keypad, but is selected automatically when CONTINUOUS is specified.

MODEs 7 and 8 cause a branch to user-written BASIC procedures at line 10000 onwards and on return the I/O are executed as in MODE 2.

MODE 9 performs various utilities with the RATE command being used to select which is executed and the WAIT specifying the duration in seconds. No MOVE commands are sent to the robot.

MODE 9 with RATE = 1 - report the current positional errors.
2 - report the current actual measured positions.
3 - report the current motor powers.
4 - perform a CONDITIONAL JUMP to the next step in the sequence as a function of the motor power.

PROCESS INPUT

Four process inputs are available (see Fig 6.1).

A zero means ignore inputs.

A positive value (1 to 4) will cause the sequence to look for that input channel to be ON, and a negative value will cause the sequence to look for that input channel to be NOT ON (see MODE 1 above).

OUTPUT

Four process outputs are available (see Fig 6.1).

A zero means "leave the status of all process outputs unchanged".

A positive value (1 to 4) switches ON the specified output channel and a negative value switches that output channel OFF.

All outputs can be on together, but not more than one can be set by a single step.

WAIT

WAIT instructs the sequence to wait for the specified number of seconds, with a maximum of 255.

WAIT is also used in conjunction with Modes 1 and 9.

JUMP

JUMP indicates the next sequence step to be executed.

A zero means that there is no JUMP, and control passes to the following step in the ordinary way.

A positive number causes the sequence to JUMP to the specified step.

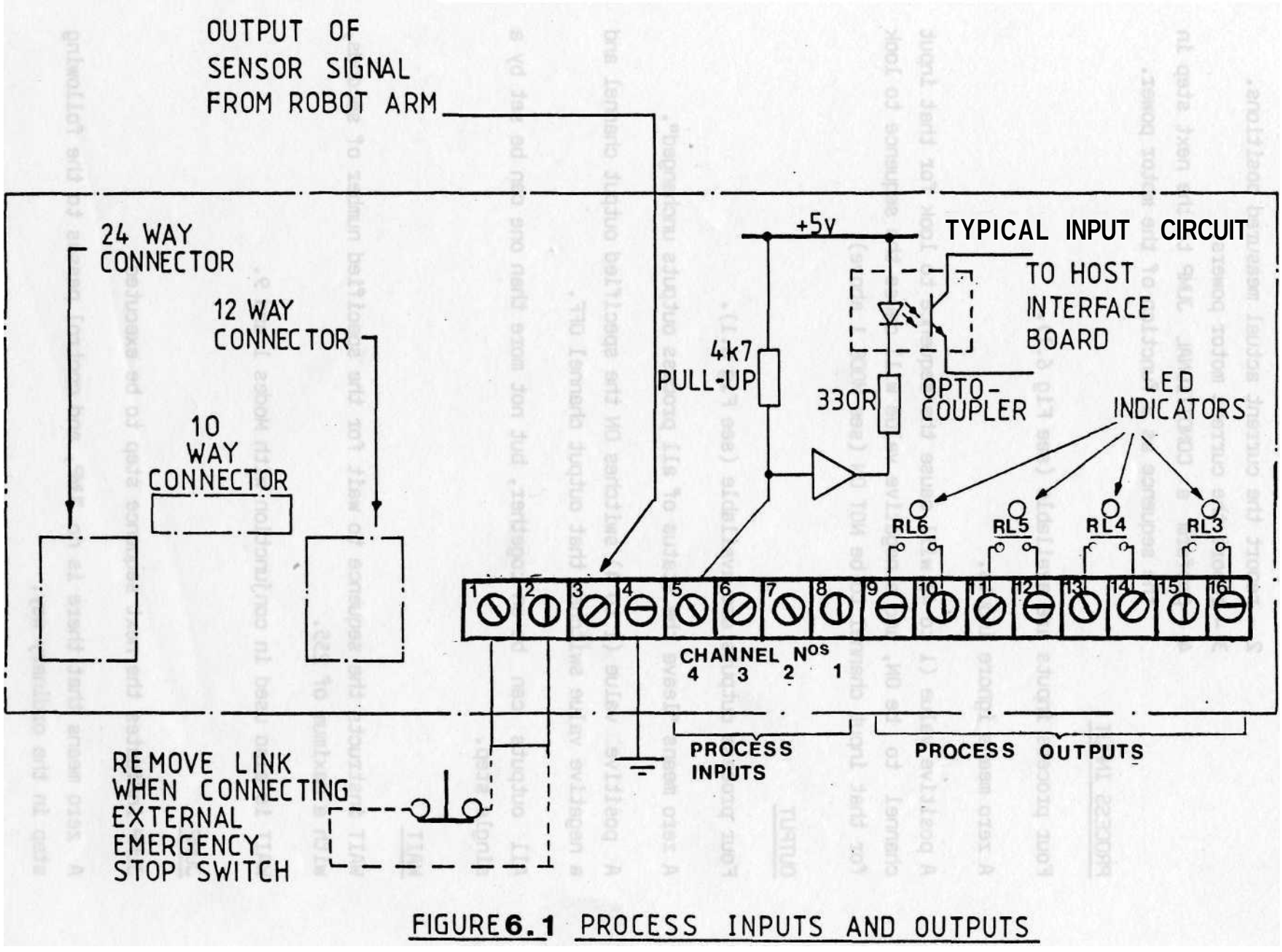


FIGURE **6.1** PROCESS INPUTS AND OUTPUTS

-1 causes return from a subsequence. Control passes to the step following the one which invoked the subsequence. Nesting is not allowed.

CONDITIONAL JUMP

When used with MODE 1 with INPUT and WAIT set, JUMP is obeyed conditionally.

SEQUENCE EXECUTION

Execution order in MODE 1 steps is described earlier in section 6 and shown in Fig 7.1.

Each MODE 2 sequence step is executed in the following order:

(a) The robot moves to the positions set in the MOTION commands subject to MODE and RATE. The gripper is set at the end of the movement.

(b) If INPUT not 0, then the sequence waits indefinitely for the specified input state.

(c) If OUTPUT not 0, then the specified output state is set.

(d) If WAIT not 0, then execution is delayed for the specified number of seconds. This wait is additional to (b) above.

(e) If JUMP is 0, >1, or -1, then control either passes to the following step, or JUMPs to the specified step or subsequence, or returns from a subsequence. (See JUMP, section 5.2.)

NOTE A <RUN> and <EDIT>

Having moved out of RUN or HOLD into EDIT to make a sequence change, on returning to RUN, the system will recommence execution from the step to which the MA2000 had last been driven, not from the step at which the last command edit was made. This is to reduce the danger of the MA2000 following an obstructed path after being commanded to drive to steps out of the desired sequence.

If the last EDIT command was a DRIVE (either to an existing position or to a new position) then that step will be taken as the continuation point upon re-entering RUN.

NOTE B REPEATABILITY

To achieve the best repeatability for any set position it is advisable to approach the position in the slower RATEs, thus the selection of faster RATEs should only be used for rapid slewing to an approximate position short of the final desired position. The following STEP should contain the final position which should be approached by a slower RATE. Inertia effects can be minimised by the selection of two or more approach STEPS, the first after the fast move at a high RATE with the following STEP or STEPs at progressively slower RATEs for final positioning.

NOTE C Use of MODE 3. "MOTION RELATIVE" or "MOVE RELATIVE"

When driving to a "blank step" which is to be a MODE 3, "MOVE RELATIVE", then the MODE must be set to 3 BEFORE the DRIVE command is given. DO NOT set the RATE at this stage.

When EDITing a sequence of MODE 3 steps always approach the step to be EDITed through the sequence, starting from a suitable absolute position. Remember DRIVEing to an existing MODE 3 step will move the robot relative to its CURRENT position.

NOTE D Use of MODE 5: SEARCH and LEARN

A step containing a safe absolute position should always be executed immediately prior to a series of SEARCH and LEARN steps, this will be used by the operating system as a default position in the event of an unsuccessful search.

NOTE E The WATCHDOG TIMER

A "watchdog" time-out circuit is fitted to the MA2000 interface board. This will automatically cause the motor controller to disable after about 3 seconds if it does not receive select signals from the computer.

NOTE F Use of MODES 7 and 8

These modes cause a jump to BASIC procedures written by the user in the procedure called "PROC user (I,J)". Normally this will consist of messages to the operator. A default procedure is at line 10000.

Calculations can be performed but CARE must be taken NOT TO AMEND any existing variables. It is strongly advised that TecQuipment be consulted about such use as THIS FACILITY REQUIRES KNOWLEDGE OF THE ROBOT SOFTWARE.

NOTE G INDEFINITE WAIT or "WAIT FOREVER"

In a MODE 1 command, if the step specified in the JUMP is the current step, then the sequence will loop until the named input is switched, and will then execute the next step. By this means the robot can be instructed to "wait forever" until a switch is pressed. For example:

```
STEP 7 )          0    1  1  0    5    7
```

A non-zero WAIT must be specified, and here a nominal WAIT of 5 seconds is used. The sequence loops through this one STEP until input 1 is set, and then the sequence proceeds to step 8, etc. This kind of logic can be extended to more complex decision loops, for example:-

```
STEP  7)          0    1    1    0    1    9
STEP  8)          0    1    0    0    0   23
STEP  9)          0    1    2    0    1    7
STEP 10)          0    1    0    0    0   84
```

Here, when INPUT 1 is set, control jumps to STEP 23, and when INPUT 2 is set, control jumps to STEP 84. These steps may initiate discrete activities and then return control to the decision loop at step 7.

In a MODE 2 STEP a non-zero INPUT command causes an indefinite wait, but when the named INPUT is set, control merely passes to the next STEP in the sequence.

7. USE OF MODES

If you feel confident about teaching the robot, it is time to learn how to use the different modes available. If you do not feel confident do not read the remainder of this chapter yet as it will confuse you.

To illustrate the different modes the following sequence will be used as the example:

The robot takes a test tube from a conveyor, places it on a balance which records its weight, then empties the contents of the test tube into a disposal unit. The robot then puts the empty test tube in a rack and takes another test tube which it puts in the conveyor. The conveyor then indexes around one space and a reagent is added to the test tube.

MODE	STEP	ACTION
2	1	move to conveyor; jump to step 50
2	2	move to balance; jump to step 56
3	3	move to rack;
4	4	search for test tube in top rack; input 3;
1	5	conditional jump 100
1	6	move clear of rack; jump to step 50
2	7	move to conveyor; jump to step *66*
	8	move conveyor; set output 1
1	9	wait for indexing signal from conveyor input 2; stop conveyor; cancel output -1
	10	start dispenser; set output 4
	11	wait till dispenser finished; wait for input 2; conditional jump 100
1	12	jump to step 1;
	50	move close to conveyor/rack
	51	move up to test tube; close gripper
	52	lift test tube clear of conveyor/rack; jump -1
	56	test tube above balance
	57	test tube on balance; open gripper; wait 5 seconds for reading
	58	positions as 57; close gripper
3	59	lift clear of balance
	60	move over disposal unit
	61	tip test tube; wait 10 seconds

```
        62      turn test tube upright; jump -1
        66      position as 52
3       61      position as 51; open gripper
        68      position as 50; jump -1
1      100      sound claxon for operator; set output 5
```

MODE 1 Control Commands Only

MODE 1 is used when only control instructions are required and the robot is to stay still. Having reached the position at which the control instructions are required press <STEP> <STEP> for a new step, then <MODE> <1> <e>. (<e> = <ENTER>). Only the control instructions will be displayed, initially set to zero. Values for INPUT, OUTPUT, WAIT and JUMP may be obeyed as described earlier. The exception is a CONDITIONAL JUMP which can be made if an input, a wait and a jump are specified in the same step.

The controller will perform the instructions according to the logic in Fig 7.1.

This means that if you want to set an output before receiving an input you must have two different steps, the first containing the output command and the second containing the input command.

If you want an output to be set on condition that an input is received then you put the output in the same step as the input.

If an input command has been given but no input is received within the specified time the sequence will jump to the step specified in the JUMP command. However the sequence will not jump until the time specified in WAIT has elapsed.

Note that if the jump is zero this will not be taken as a jump to step 0 but will be ignored and the following step will be executed.

In the example of the test tubes, once the weighing has taken place the conveyor must be indexed one place and the reagent dispensed. So after the test tube has been replaced in the conveyor the sequence enters MODE 1.

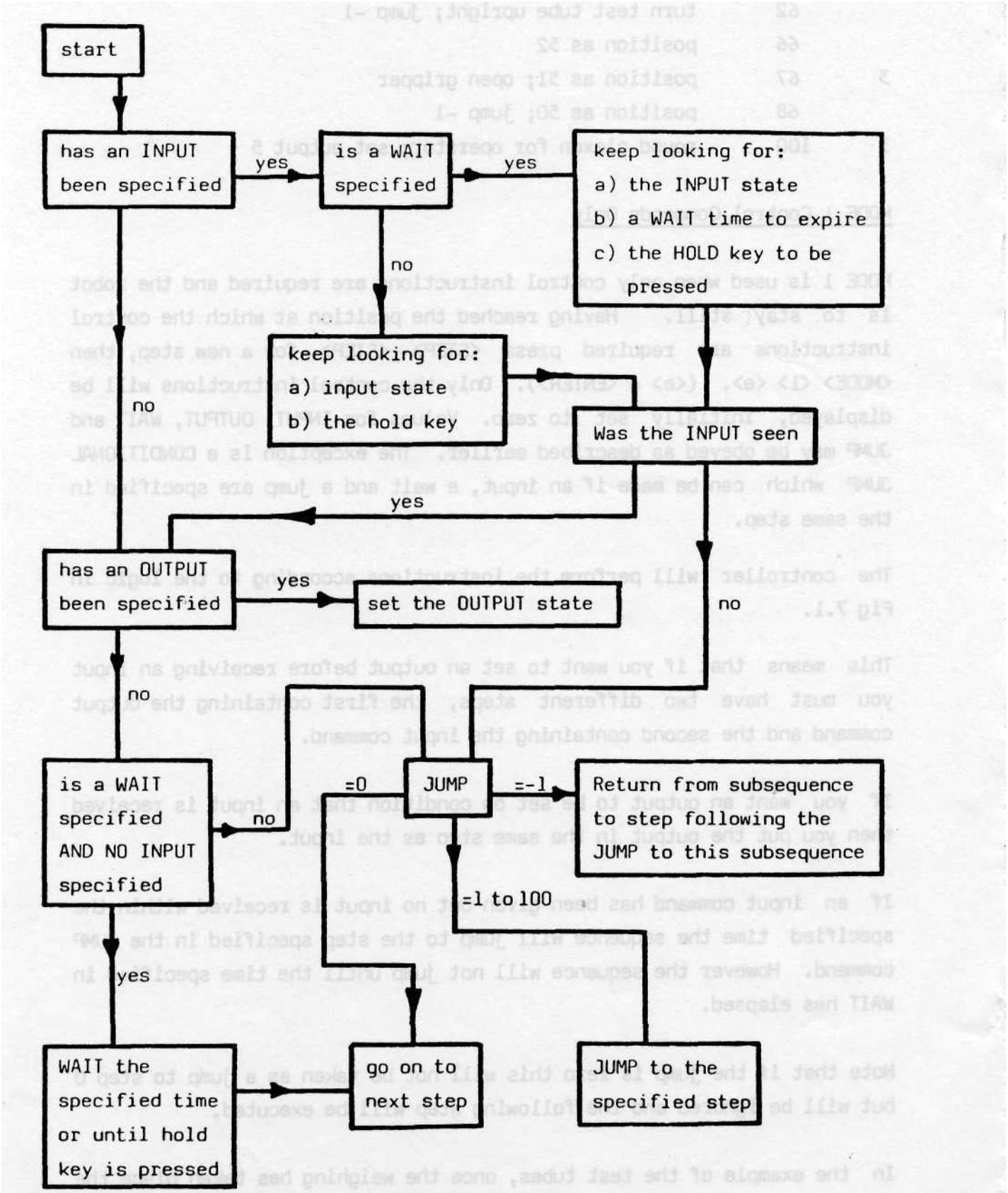


FIG 7.1 Mode 1 Logic

Note: This logic is followed in Mode 1. Use of Input in other Modes does not give conditional jumping.

First an output is set to move the conveyor around:

	RATE	MODE	IN	OUT	WAIT	JUMP
Step 8	0	1	0	1	0	0

Next the conveyor is prevented from indexing around and around; the position signal is examined and then the conveyor is stopped:

Step 9	0	1	2	-	1	0	0

Now to operate the dispensing unit:

Step 10	0	1	0	2	0	0

Because of the nature of the dispensing unit the robot controller waits 20 seconds for an input from the dispensing unit to say the reagent has been added:

Step 11	0	1	1	0	20	100

If the input is not received within 20 seconds the sequence will jump to step 100 where the output will set off an alarm to tell the operator the system has malfunctioned. If the input is received the sequence goes on to step 12 which returns to the beginning of the sequence.

MODE 2 Move to Absolute Positions

Since you have been using MODE 2 until now it needs very little explaining. MODE 2 makes the sequence move through incremental positions with speeds as set by RATE until it reaches the final positions contained in the step.

MODE 3 Move Relative to Last Position

MODE 3 steps are most usefully used within subsequences.

MODE 3 enables the robot to perform a subsequence in different places with the operator only having to teach the robot the subsequence once and then instructing the robot at which places it is to perform the subsequence. In our example the robot performs 3 subsequences - the picking up of the test tubes from the conveyor and rack; the weighing sequence; and replacing the test tubes.

The step in which the jump-to-subsequence command is given is the all

important step. The position of such a step is the position from which the subsequence motions increase or decrease. If this initial step is not taught accurately all attempts at subsequences will fail.

If you are about to program a part of a sequence that you know is going to be a subsequence, first decide on a step number much further on in the sequence which will be the subsequence. In our example step 100 has been chosen as the start of the pick-up subsequence. In the step at which you are holding put in <JUMP> <1> <0> <0> <e>. Now press <STEP> <1> <0> <0>. Do not press <STEP> <STEP>.

BEFORE PRESSING THE DRIVE KEY PRESS <MODE> <3> <e>.

Do not set the rate yet, in fact make sure the rate is 0 at this point. The step co-ordinates will show absolute positions whilst in DRIVE. You can now press <DRIVE> and the motion keys to make up your subsequence going from step to step as normal.

When a MODE 3 step is learnt, the display positions are automatically converted from the actual robot position to relative positions; the number 500 is used as the datum (for example 430 indicates that the step will decrease the angle of the joint by 70 units, 500-430). MAKE A NOTE OF THE CO-ORDINATE NUMBERS which the robot is moved to, this saves time later on. When you have finished your subsequence press <JUMP> <-> <1>. This will make the sequence go back to the step following the step which contained the jump command for the subsequence.

Now you can carry on teaching the sequence in MODE 2. When you reach the next position where you want the subsequence to be performed you just enter the JUMP command.

If you want the robot to perform the subsequence in the same place as before, the step which includes the jump-to-subsequence command must have the same co-ordinates as last time. In the example the weighing subsequence occurs twice at the same place at both steps 2 and 9. So to make sure step 9 is the same as step 2, step 9 was taught its position by pressing <COPY> <2> <e> <2> <e> <9> <e> <9> <e>. This copied step 2's position into step 9. (Copying a single step is a little cumbersome

but this format allows large blocks of steps to be moved about in one operation). Then the jump to step 56 was entered. At step 62 the sequence will return to step 10.

However if the robot is to perform the subsequence at a different place, as in the picking up and replacing subsequences, it is important that the step containing the jump is at the correct place for the subsequence to take place. This is where the co-ordinate numbers you have written down come in useful. The numbers are either more than 500 or less than 500. The difference between the number and 500 is the number of increments the motion will move from where it is NOW.

Editing MODE 3 steps is tricky. To edit a series of MODE 3 steps you must start from an appropriate absolute (MODE 2) step and drive to alter and re-learn the MODE 3 steps in their correct sequence. Great care must be taken as when you are "driving" to the step the robot will move relative to where it is now - even if it was already "at" the MODE 3 step.

So to make sure your robot is in the right place move it using the key pad to the first position you want it to go to in the subsequence. Since you know the number of increments moved between this place and the step which is to contain the jump, subtract or add the number of increments applicable to each axis (do this on paper it is easier) then move the robot motions to the results you have just worked out. This should be done very slowly and carefully as it is important that the step before the subsequence is accurate or else the subsequence will not be performed accurately.

Picking up of the test tube in the example can be equated to this. Step 1 was to move to the conveyor ready for step 100 which would take the robot just clear of the conveyor ready to pick up the test tube.

Step A must be exactly the same distance from the rack as step 1 was from the conveyor.

Caution When using subprograms and relative motion, there are two very easy traps to fall into:

(i) Never, ever, press <RUN> after creating a subprogram. If you do the Robot will move relative to the final position in the subprogram which will almost always cause excessive movement or collision. After all, you have designed your subprogram to arrive at its final location, not depart from it.

(ii) Never, ever, put a subprogram in a loop. The relative increments will add together and drive the Robot out of limits.

MODE 4 Search

MODE 4 enables the robot to perform searches. To do this requires some form of sensor. In the example a simple micro switch is fitted inside at the back of the gripper. When using MODE 4 you teach the robot the end position of the search and tell it what input to expect. If the input is found the robot will stop and go on to the next step as a result of the input being found. It will not continue to the position given in the step if an input is found. It should be noted that if the distance to be searched is not small the robot will move in an arc rather than a straight line, particularly if only one motion is to be moved.

In our example the robot requires an input from the gripper switch at step 3 to locate a test tube. The robot will move from its present position to another position further along the rack which has been specified. This step was taught in MODE 4 so that if the robot controller receives the input before reaching the specified position further along the rack, the robot will abandon its search and the sequence will go immediately to the next step. The search can be followed by a conditional branch step also looking for input 3 (MODE 1). If input 3 is not received the sequence jumps to step 100 to alert the operator.

8. PROGRAMMING GUIDELINES

Assuming that you are now fairly conversant with the keypad, the screen display and the various robot motions, you will be keen to know how, quickly and efficiently, to teach the robot a sequence that will perform your task. There are a number of hints and techniques that can be passed on in the light of experience, most of which are common sense.

"Keep the limitations of the robot in perspective"

The "repeatability" of the robot at full reach is approximately +/- 2mm. This means that at any of the step positions there is a possible variation of up to 4 mm. Thus if the robot is to pick or place anything it is vital to take this into account. For example, in picking up an object, the open gripper jaws should clear the part by at least 2 mm in order to clear it every time. Likewise when placing an object "down" or "in", similar clearances must be provided. An alternative line of attack is to use taper fits, bevelled edges etc, although this is only applicable if modifications are possible to the equipment with which you are working.

"Keep the mechanical arrangements simple"

Although not strictly a programming guideline it is worth repeating the recommendation made in the installation section to keep the movements of the robot as simple as possible. This can be achieved by arranging pick-up and put-down locations to be easily accessed by the movement of one or perhaps two motions. For example, to pick up a can of soup from a flat surface it should be possible to move the robot to the pick up position by lowering the shoulder only. Likewise loading the chuck of a lathe should only need the robot to swivel on its rotate motion.

"Use the cart-before-the-horse technique"

When teaching a "PLACE" sequence it is best to start with the robot in the placing position. Give this a step number 5 or 10 higher than the beginning of the sequence. Then move BACK one step and if possible use just one motion to move the robot to a position where the gripper is clear of the "place" position and learn this step. Then if the robot is

unlikely to clout anything between this and the park position, the step before this can be the park position. If the robot needs to avoid fixed objects between the gripper "clear" and the park positions then you will obviously need more steps to have a tighter control over its trajectory.

If you teach a sequence in the opposite order (i.e. the order in which the robot will move and thus the order you would logically expect) you will find that you will move some joints unnecessarily as you approach the part.

"Avoid re-teaching"

It is possible to make considerable savings in programming effort if you appreciate that often you can use the same step a number of times.

Using the previous example in which we were placing a part, we can illustrate this point. We will assume that the previous part of the sequence used steps 20, 21 and 22 (20 being the park position). Having placed the part, it may be that you wish to move the robot back to the park position. The only difference between the approach and retract needs to be that the gripper will be open in the latter. If step 22 places the part in the correct position and opens the gripper then step 23 can be the same as step 21 and step 24 the same as step 20.

The <COPY> command can be used to copy step 20 into step 24 and step 21 into step 23 (see para 5.2 for details).

"Use subsequences wherever possible"

As with conventional programming, the use of subsequences can greatly reduce the programming effort. If at some points in your taught sequence you use the same set of steps, then it is a good idea to parcel these moves into a subsequence. For example, if you were handling test tubes in which a washing cycle was repeated a number of times, the wash cycle could be made into a subsequence. At the relevant points in the sequence you would include a JUMP instruction to the starting step of your subsequence. To return from a subsequence a command JUMP -1 is given which returns the sequence to the step following that which contained the JUMP-to-subsequence instruction.

This is all well and good when you are using MODE 2 (or absolute co-ordinate) program steps. Even better use of subsequences is made when you teach the subsequence in MODE 3 (or relative co-ordinate) steps.

If you have to pick up a part from a number of locations then it is possible that the majority of the joint movements are common to each pick-up operation with only the rotate angle differing. The common joint movements would then be taught in MODE 3 and stored as a sub-sequence. In the sequence the robot would be moved only in the rotate axis to the correct angle for each part and then a JUMP would be made to the pick-up subsequence.

If you change the position of any of the parts you will now be able to correct the sequence by changing one step only; the step from which you call the subsequence.

Remember, however, that subsequences cause movements in "Robot Joint Space", not in cartesian space.

9. PROCESS INPUTS AND OUTPUTS

Refer to figure 2 for the connections to the four process inputs and four process outputs.

Process Inputs

To energise an input a connection from an input terminal to the ground terminal is required.

A switch, open-collector transistor or opto-sensor may be used providing the load current of 10 mA is obtained.

If additional TTL logic is fitted on the inputs, it should be capable of providing the required input current of 1.5 mA.

The inputs are opto-isolated from the host computer signals to ensure that external noise and earth loop volt drops do not cause a mal-function.

ON NO ACCOUNT MUST UNAUTHORISED CONNECTIONS BE MADE TO THE MOTOR CONTROLLER OR HOST COMPUTER CIRCUITRY

Process Outputs

Each output consists of a miniature relay which can be energised under program control. The contacts are rated at 24 V d.c. at 1 amp RESISTIVE load.

10. FAULT INDICATIONS

Certain conditions will cause the audible alarm on the MA2000 computer to sound.

(a) The most likely cause of the alarm sounding during the execution of a program step is that too high a rate has been selected, causing at least one motor to be running at its maximum speed. This would not cause any damage but is not advisable. A lower rate setting for that step should be selected.

(b) If the MA2000 computer detects invalid status information from the micro-controller, indicating that the micro-controller is not operating correctly, the alarm will sound and

CONTROLLER STATUS INVALID

will appear on the screen. Press the RESTART button to reset the system. Hold the button in for a period before releasing it to allow the hardware resets to fully operate. Check also that the computer select switch on the robot controller is in the correct position (B for IBM working).

(c) If during the execution of a programmed step or during drive either the emergency stop is pressed or AUTO is de-selected, then the alarm will sound and the robot path frozen. Releasing the emergency stop or returning to AUTO will allow the path to be resumed.

11. DEMONSTRATION SEQUENCE

STEP 0)		7	2	0	0	0	0
	500	400	100	500	500	500	0
STEP 1)		7	2	0	0	0	0
	500	500	500	500	500	500	0
STEP 2)		6	2	0	0	0	0
	800	800	400	400	500	500	1
STEP 3)		4	2	0	0	7	0
	250	250	200	400	500	500	0
STEP 4)		7	2	0	0	0	0
	800	800	500	400	500	500	0
STEP 5)		6	2	0	0	0	1
	400	200	500	400	500	500	1
STEP 6)		5	2	0	3	5	0
	500	500	500	500	500	500	0
STEP 7)		7	2	0	-3	0	0
	990	800	400	400	500	500	0
STEP 8)		0	1	3	0	5	6
STEP 9)		8	2	0	0	0	0
	800	800	450	400	500	500	0
STEP 10)		6	2	0	0	0	6
	10	250	400	500	500	500	0
STEP 11)		7	2	0	0	0	14
	500	500	500	500	500	500	0
STEP 12)		6	2	0	0	0	14
	800	350	450	400	500	500	0
STEP 13)		7	2	0	0	5	11
	400	400	400	400	500	500	0
STEP 14)		3	3	0	0	0	0
	500	350	450	400	500	500	0
STEP 15)		3	3	0	0	0	-1
	420	580	380	500	550	500	0
STEP 16)		6	2	0	0	0	0
	500	500	500	500	500	500	0
STEP 17)		0	1	1	0	0	0

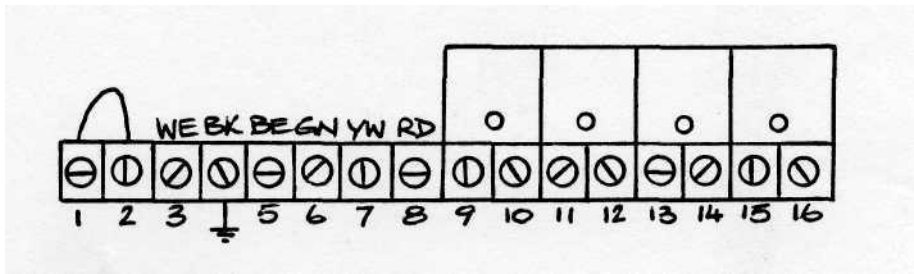
STEP 18)		4	4	2	0	0	0
	200	400	200	400	500	500	0
STEP 19)		3	4	3	0	0	16
	800	600	550	400	600	500	0
STEP 20)		7	2	0	0	0	0
	500	500	500	500	500	500	0
STEP 21)		2	5	4	0	0	26
	200	400	400	400	500	500	0
STEP 22)		3	5	3	0	0	27
	800	400	400	400	500	500	0
STEP 23)		3	5	1	0	0	28
	200	400	400	400	500	500	0
STEP 24)		4	5	2	0	0	29
	200	400	400	400	500	500	0
STEP 25)		7	2	0	-4	0	0
	500	500	500	500	500	500	0
STEP 26)		5	2	0	0	0	0
	0	0	0	0	0	0	0
STEP 27)		7	2	0	0	2	0
	0	0	0	0	0	0	0
STEP 28)		5	2	1	0	2	0
	0	0	0	0	0	0	0
STEP 29)		4	2	0	4	0	0
	0	0	0	0	0	0	0
STEP 30)		7	2	0	0	5	20
	500	500	500	500	500	500	0

M A 2 0 0 0 p B U T T O N B O X

CONNECTION AND USE

The TecQuipment robot control button box, MA2000p, which is often referred to as the 'decision box', can be used to control running sequences and to simulate external switches. The box contains four normally open (push-to-make) switches, labelled 1 2 3 +4, and one normally closed (push to break) switch labelled -4. A toggle switch is used to select either button +4 or button -4.

The cable attached to the box has six cores, red (RD), yellow (YW), green (GN), blue (BE), black (BK), and white (WE). These should be connected to the rear of the robot micro controller/interface as follows:



Terminals 1 and 2 form the emergency stop loop.
Terminal 3 is the return from the signal wire on the robot end effector.
Terminal 4 is the ground or earth.
Terminals 5 through 8 are INPUT channels 4,3,2,1 respectively.
Terminals 9 through 16 are the four OUTPUT pairs.

When the MA2000 sensing grippers are used, the toggle switch on the decision box should be switched to -4, and the INPUT value -4 should be used in sequence steps to detect an object within the gripper (see MA2000a Experiments Kit Manual, section 7,2, but note that the decision box takes the place of the wire from terminal 3 to terminal 5).

With the toggle switched to +4, the button box can be used to control a decision loop, as in Note G, p.28 of the MA2000 manual "The Open University Robot". This style of sequence control is used in SEQUENCE 1234 supplied with MA2000 software Iss.3.80 onwards.

An alternative use in simpler sequences is to halt the execution at certain critical phases, by introducing an INPUT request to be satisfied by the operator via the button box when he is sure that it is safe or useful to proceed. For example, putting INPUT 1 in STEP 1 of a sequence which cycles (i.e. last step has a JUMP 1) will mean that the sequence will halt after every cycle until button 1 is pressed.

AAW/1285/i