A100/200 Series SMALL INDUSTRIAL ROBOT SYSTEM

TUTORIAL MANUAL

If the display shows different numbers than these, you should find out what is in the memory.

STEP 3) Look at the contents of the Program Table (directory) with the >>**DIR** <cr> command.

This shows the programs by name and size in bytes. Pick a program to list and display it on the screen using the **LISTP** command.

STEP 4) Type >>**LISTP PROG_NME**<cr>. If the program fills more than one screen, the screen output pauses until you press a key.

Note how you had to type the entire LISTP command. This is because there are several other commands with the same first four letters (LISTV, LISTL). The syntax builder completes the command only when enough letters to make the command name distinct are entered.

If there are programs in memory, there most likely will also be locations in the location table.

STEP 5) These can be seen by typing >>**LISTL** <cr>.

Observe that the locations are stored by their names and their cartesian coordinates. The X, Y, and Z dimensions are in inches, (or in millimetres) while Yaw, Pitch and Roll are in degrees. The robot's coordinate system is explained in Chapter 2-2 of the RAPL-II manual.

3-2 BASIC MOTION COMMANDS

There are several motion commands in the RAPL-II language. You have already used the JOG, READY, ALIGN and JOINT commands. Others include MOVE, APPRO, DEPART and MOTOR. The MOTOR command causes an incremental motion of one motor by a specified number of pulses. It is not often used in programming the robot.

The MOVE, APPRO and DEPART commands are used often. The APPRO and DEPART commands allow the robot to move to and from points near a programmed location. Try approaching the defined point: POINT_1.

STEP 1) Type >>**APPRO** LOCATION : **POINT_1**, BY A DISTANCE OF :**2<cr>**. The robot will move to a point 2 inches back from POINT_1. Technically, this point is 2 inches away from the specified location POINT_1 along the tool axis. This is the "direction" in which the tool, the gripper in this case, points. At POINT_1, the gripper points vertically downward so the approach point is directly above the taught location.

STEP 2) Now move the robot to POINT_1 using the MOVE command. Type >>**MOVE** TO LOCATION : **POINT_1<cr>** The robot moves down to POINT_1.

From here, you can try the DEPART command. Actually the DEPART command will work from any robot location (as long as it does not cause a joint to go out of range).

STEP 3) Type >>**DEPART** BY A DISTANCE OF:**2<cr>**. The robot will depart 2 inches along the tool axis.

STEP 4) Now issue a JOINT command to change the direction of the tool axis: >>**JOINT** JOINT #: **4**, BY DEGREES: **20<cr>** and try another DEPART by 2 inches. Note that the direction of departure has changed to the new direction of the TOOL axis.

You can also DEPART in the negative direction which will move the robot towards the point. <u>Be careful not to drive the gripper into a stationary object by doing this.</u> Because of the relative nature of the DEPART command, it can be used for accurate final positioning during location programming.

CHAPTER 5 - MORE ADVANCED PROGRAMMING TECHNIQUES

5-1 COUNTERS AND VARIABLES

All RAPL-II commands and their descriptions are listed in CHAPTER 4 of the RAPL-II Programming Manual. With these descriptions and the syntax builder (Help mode), you should be able to implement these commands without difficulty. From here on, this manual will be less explicit when instructing for command entry. New commands will be shown in detail but commands you have seen before will not be shown explicitly.

STEP 1) Use the COPY command to copy the program PICK to PICK1 so that you can make changes to the program without affecting what you have already done. Type >>COPY OLD PROGRAM NAME: PICK, NEW PROGRAM NAME: PICK1<cr>

STEP 2) Try implementing a counter in the PICK1 program. To do this add the following lines. Remember how to edit a new program: use the EDIT command followed by the name of the program you wish to edit, in this case PICK1. You can use the insert command within the editor, or leave the editor and use the PLE mode to insert the lines:

```
5 ! COUNT = 0
35 ++ COUNT
```

Line 5 initializes a variable called "COUNT" and sets it to zero. The !, is the assignment command in RAPL-II (assigns a value to a variable). Because of the looping back to line 30 this is done only once at the start of execution. Line 35 increments COUNT by one each time through the loop. The ++ command will increment the variable by 1 (same as COUNT = COUNT +1). Similarly the -- command will decrement a variable by 1 (same as COUNT = COUNT - 1). This statement could be placed anywhere inside the loop to be effective.

Variable names can be up to eight characters in length. Any numeric or alphabetic characters are legal, but the first character may not be a numeric character (see RAPL-II Programming Manual, Section 1-9 for more information).

Variables are useful in programs for counting loops. To let PICK1 get 5 parts and then stop, add the following line:

```
112 IF COUNT EQ 5 THEN 120
```

This will check the value of COUNT each time through the loop. When it has picked and placed 5 parts, COUNT will equal 5 and program flow will jump to the STOP line before it reaches the GOTO 30.

ALL FACTORY COURSES FEATURE

Location:
All courses are taught in a fully equipped training room at our factory, or, a comparably equipped conference room in a local hotel. Some courses may require visiting the factory (transportation will be provided).

Catering:
All courses include a catered lunch and "bottom-less" coffee or tea. Although breakfast and dinner are not included, a local restaurant guide is available.

Orientation:
If your order is received 2 weeks before the course an orientation kit will be mailed to you. This kit contains information that will help make your trip as enjoyable as possible, and includes tourist brochures/maps for the city and province.

If you place your order less than 2 weeks before the course, you should contact the training department for special arrangements.

Lodgings:
A list of local hotels is included in the orientation kit.

TERMS AND CONDITIONS

1. All courses require payment in advance in order to reserve a place at a particular course. An invoice will be sent immediately upon booking and will be payable upon receipt. Confirmation of attendance at the course will be made upon receipt of full payment.

2. The full name and address of students taking the course should be sent to CRS 30 days before the course is scheduled to begin. This will allow CRS to send orientation kits directly to students 2 to 3 weeks before the course begins.

3. Cancellations made 30 days or more before the scheduled course date will be 100% refundable.

   Cancellations made 15-30 days before the scheduled course date will not be refunded. However a 100% credit will be given, if applied to an equivalent CRS PLUS Training course held within the next 12 months.

   Cancellations made less than 15 days before the scheduled course date and "No-Shows" will not be refunded. However a 75% credit will be given, if applied to an equivalent CRS PLUS Training course held within the next 12 months.

   "No-Shows" or cancellations made by distributors or their reps less than 30 days before the scheduled course date will result in a 50% service charge.

4-5 TRACE MODE (Continued)

Each line has the following components:

[00]	This is the nesting level. The main program executed from the RUN command is level 00. The first subroutine called is level 01. If that subroutine calls another, the new subroutine is level 02, and so on. You will be formally introduced to subroutines in an upcoming section.
PICK	The name of the program or subroutine being executed.
#00010	The line number
OPEN	The contents of the line.

Note that flow passes through lines 10 and 20 only at the first run through. From then on it flows from lines 30 to 115 and back to line 30 again. This pattern will be repeated indefinitely.

STEP 3) Use <Ctrl-A> to abort the program.

STEP 4) Type >>**DISABLE TRACE**<cr> to turn off the Trace mode.

If you wish to "fine tune" the gripper's location, the JOG command can be useful.

STEP 5) Measure the distance from the current position to the desired position in inches in three directions (the robot's XYZ coordinates).

STEP 6) Type >>**JOG** DX,DY,DZ : **1,0,.5<cr>** (the increments of distance in the three directions in turn). The robot will move the distances desired in a straight line.

How to store a desired location:

If you are satisfied with the location and wish to save it to memory, use the HERE command to store it:

STEP 1) Type the HERE command: >>**HERE** LOCATION : **POINT_1<cr>**.

STEP 2) Verify that the location has been stored by a **LISTL** command. Type >>**LISTL<cr>**. You should see POINT_1 somewhere in the location table.

STEP 3) Return the robot to the READY position.

In the task you are programming, you want to pick up a part at POINT_1, and put it down at a location a few inches away, POINT_2. To teach the robot POINT_2, you could repeat the above exercise of moving the arm, aligning it, fine tuning it and teaching it, but that would be inefficient. Try another method.

STEP 4) RAPL-II lets you enter a location equal to another one using the **SET** command. Type >>**SET** LOCATION = LOCATION<CR> **POINT_2** = **POINT_1<cr>**. Now the location table contains two locations POINT_1 and POINT_2 with identical coordinates (Another way would have been to just enter
>>**HERE** LOCATION : **POINT_2<cr>** while still at POINT_1).

STEP 5) Since you don't want the robot to put the part back down where it got it from, you must SHIFT the location called POINT_2 away from POINT_1. Type >>**SHIFT** LOCATION : **POINT_2** DX,DY,DZ :**0,8,0<cr>**. This has the effect of moving the location POINT_2, 8 inches away in the positive Y direction from POINT_1.

STEP 6) Confirm this using the LISTL command. In the table that is displayed, note that the Y coordinate of POINT_1 is exactly 8 inches different from the Y coordinate of POINT_2. The value at POINT_1 should be -3.7366 and the value at POINT_2 should be +4.2633. The difference between these values is +7.9999

ie. 4.2633 - (-3.7366) = 7.9999

You are now ready to use these locations to demonstrate some simple motion commands.

1-7 ENTERING COMMANDS (Continued)

If a mistake is made when entering a command, there are two possible outcomes:

1) A letter or sequence of letters is typed which identify another command which RAPL-II accepts. This puts you into the potentially dangerous situation of having entered a command you do not want, or

2) you type a sequence of characters which cannot possibly lead to a legal RAPL-II command. In this case RAPL-II aborts the command and displays an error message (usually ERROR #12, a COMMAND ERROR). If you went on and typed more characters after the error, you may have several such errors or another "type 1)" mistake.

In both cases, an <ABORT> sequence may be needed to restore the RAPL-II prompt and prepare you for another try.

1-8 SYSTEM CHECK OUT

There are several commands which allow you to check the status of the robot system and user RAM memory. The first is the **STATUS** command.

STEP 1) Type >>**STATUS** <cr> at the prompt.

The display will be filled with information about the current status of the robot system. The information displayed is described further in the RAPL-II manual under the **STATUS** command.

Next, look at the amount of unused user memory with the **FREE** command.

STEP 2) Type >>**FREE** <cr>

The display shows the amount of memory remaining. An empty robot with a 16K memory, with default memory allocation, will show: 68 variable, 68 locations, 16 programs, and 3856 bytes of free program memory.

REV.	REVISION HISTORY	DATE
-001	Original Issue	2/86
-002	RAPL Version 3.30 coverage	9/86
-003	RAPL Version 4.11 coverage	2/89
-004	RAPL-II and A100/200 support	8/90

Additional copies of this manual, or other CRS Plus literature, may be obtained from:

CRS Plus Inc.
830 Harrington Ct.
Burlington, Ontario,
CANADA.
L7N 3N4

Telephone: (416) 639-0086
Facsimile: (416) 639-4248

CHAPTER 2 - STARTING UP THE ROBOT

2-1 ARM POWER

Up to this point, the robot arm has been Limp (unless the arm is equipped with brakes). Before the controller can move the arm, the power to the motors must be turned on. To turn on the arm power:

STEP 1) For a robot not equipped with brakes, grasp the arm by its end effector and lift it to approximately the READY position (see Figure 2-1). Press the ARM POWER switch on the lower right of the front panel of the controller; the light in the push button will come on. Notice that the robot arm is now locked in position.

STEP 1) For a robot equipped with brakes, press the ARM POWER switch on the lower right of the front panel of the controller; the light in the push button will come on. As soon as it has, the brakes disengage and the action of supporting the arm is taken over by the motors.

As soon as arm power is applied, the controller is controlling the position of the arm. The computer is constantly issuing commands to each motor to move the arm. The mechanism (including the motor) which follows the controller commands is called a servo (short for "positional SERVO-mechanism"). The servo monitors the position of the motor shaft, and attempts to correct any difference between the commanded position and the motor's actual position. The operation of the servo system is described in more detail in the Technical Manual.

Although the arm looks stationary, it may in fact be moving in tiny steps up and down. Because of gravity the controller must always exert a force on the arm upwards to keep it still. This action is called "servoing".

It is important to realize that at all times when the arm power is on, the position of the robot arm is controlled by the system computer. If the computer tells the arm to move, it will do so without warning! This is particularly true when the system is executing a program. In many robot tasks, the machine may stand still for an extended period of time before suddenly moving.

⚠ PERSONNEL MUST NOT ENTER THE ROBOT WORK SPACE WHEN THE MACHINE IS UNDER PROGRAM CONTROL.

An interruption of the servo system may cause the robot to move in an unpredictable way. Under normal operation this will not happen, but, in the case of component failure it could.

As with any automated equipment, you must be prepared to react to an emergency situation. Should the robot, or any component of the work cell, do something completely unexpected you must know how to react to it. There are two ways you can stop the robot, and one way it can stop itself.

3-3 GRIPPER COMMANDS

With a few more commands, you will be equipped to program the robot to do a simple "pick-and-place" routine. These commands are the Gripper commands.

Pneumatic Gripper:

If you have an air gripper connected to the robot and hooked up to an air supply then you may use the OPEN and CLOSE commands. The CRS Plus robot comes equipped with a four-way pneumatic valve. This will pressurize the gripper in both the open and close directions.

Pneumatic Gripper (continued):

STEP 1) To close the gripper type >>CLOSE<cr>, for CLOSE.

STEP 2) To open the gripper type >>OPEN<cr>, for OPEN.

The gripper closing force depends on the air pressure used. The toggle switch marked "GRIP" on the Teach pendant opens and closes the pneumatic gripper when the controller is in MANUAL mode.

Servo Gripper:

If a servo gripper is used, the OPEN and CLOSE commands will also open and close the gripper. However, along with the command, you must specify a force, as a percentage of full force.

STEP 1) Type >>OPEN GRIPPER OPENING FORCE(%): 40<cr>, to open the servo gripper at 40% of full torque.

STEP 2) Type >>CLOSE GRIPPER CLOSING FORCE(%): 40<cr>, to close the servo gripper at 40% of full torque.

The GRIP command is also used when controlling the servo gripper. This command controls the gripper in the position mode positioning the gripper fingers to the specified opening in linear units (inches or millimetres). When commanded with GRIP, the gripper operates at maximum force to minimize motion time. DO NOT USE this command to grip an object as it will be at maximum force and could damage the part! Also leaving it in that state for a prolonged period WILL damage the gripper! (See warning below.) With nothing between the gripper fingers, try out the GRIP command:

STEP 3) Type >>GRIP GRIPPER OPENING (INCHES/MM): 1<cr>. The finger platforms should move to an opening of 1". If using the system in metric unit mode, use >>GRIP 30<cr>.

5-1 COUNTERS AND VARIABLES (continued)

Variables in a program can also be used to shift locations (see program PALLET). If for example you wish to stack or load and unload palletized parts, then a SHIFT command with a variable can become extremely useful. As an example, if the 5 parts were at five locations, differing by 0.75 inches, the PICK location would have to be shifted as in the example program. Note that a location called PICK_REF must be defined at the original location of PICK.

NOTE: This program is intended as an example, you do not have to type it in.

```
PROGRAM: PALLET
20 ; Initialize the program:
30 ;
100 OPEN
120 ! COUNT = 0
130 SET PICK = PICK_REF
140 ;
150 ; Start of loop:
200 READY
210 ;
220 ; Determine the new position of PICK-up:
240 ! OFFSET = COUNT * 0.75
250 SHIFT PICK BY OFFSET,0,0
255 ! COUNT = COUNT + 1
260 ;
270 ; Pick and Place routine:
300 APPRO PICK,2
310 MOVE PICK
320 FINISH
330 CLOSE
340 DEPART 2
350 APPRO PLACE,2
360 MOVE PLACE
370 FINISH
380 OPEN
390 DEPART 2
400 ;
410 ; Check counter and branch:
430 IF COUNT EQ 5 THEN 600
440 GOTO 200
500 STOP
$
```

Notice the use of comment lines in the program. These will help you to understand what you did if you look at the program sometime in the future.

APPENDIX A - CRS PLUS FACTORY TRAINING COURSES

Robot Introductory course SRS-M1/BASIC:

This course is designed to be an introduction to CRS Plus robot systems while providing a solid understanding of M1 series installation and operation. Key topics covered include: system components, installation, operation, routine maintenance, diagnostics, manual mode operation, program mode operation, trouble shooting, servo operation, and robot calibration. After completing this course the student will be able to install, test, operate, and maintain any M1 series industrial robot. This course includes these manuals: RAPL, Technical, Tutorial, Palletizing, and PATH.

Course Length	:	3 days
Prerequisites	:	None. Computer experience an asset.

Robot Advanced course SRS-M1/ADVCE:

This is an intensive hands-on course concentrating on advanced system programming of the M1 series motion controller using its advanced, built-in, operating system: RAPL Simple examples are introduced and then developed into working applications. CRS Plus automation experience is used to teach students how to predict, and account for, "real world" complications. Key topics covered include: structured programming, palletizing, constant velocity motion, operator safety, and error recovery. After completing this course the student will be able to design and write complete robot application systems for the M1 series industrial robot. This course includes these manuals: RAPL, Technical, Tutorial.

Course Length	:	2 days
Prerequisites	:	SRS-M1/BASIC. Structured programming an asset.

Service Training SRS-M1/SERVC:

This course covers maintenance, service, and internal operation of the M1 series industrial robot. Original blueprints, schematics and service manual are used to explain every detail of design, operation, and repair. Key topics covered include: maintenance, trouble-shooting, calibration, servo operation, electronic hardware function, mechanical assembly function, and field service procedures. After completing this course the student will be able to diagnose and repair the SRS M1A industrial robot system. This course includes these manuals: RAPL, Technical, Tutorial, and Service.

Course Length	:	3 days
Prerequisites	:	SRS-M1/BASIC, and either Electronics or Mechanical knowledge.

4-4 CREATING A "LOOPING" TASK (continued)

```
>>LISTP<cr>
PROGRAM: PICK
10 OPEN
20 READY
30 APPRO PICK,2
40 MOVE PICK
45 FINISH
50 CLOSE
60 DEPART 2
70 APPRO PLACE,2
80 MOVE PLACE
85 FINISH
90 OPEN
100 DEPART 2
110 READY
115 GOTO 30
120 STOP
$
```

Notice that the STOP command will never be reached since program flow will jump to line 30 before the STOP is executed.

4-5 TRACE MODE

If you have made some error in the program logic, one way to look into it is to use the Trace mode.

STEP 1) Before running the program, type >>**ENABLE ITEM NAME OR <CR>: TRACE<cr>** to enable the tracing of the program.

STEP 2) Type >>**RUN** <cr>.

Running the program now will cause the robot to run the pick-and-place routine continuously. As each line is executed, the screen displays information about that line. You may see the following after the RUN command:

```
[00]PICK#00010 OPEN
[00]PICK#00020 READY
[00]PICK#00030 APPRO PICK,2
[00]PICK#00040 MOVE PICK
[00]PICK#00045 FINISH
......
[00]PICK#000110 READY
[00]PICK#000115 GOTO 30
[00]PICK#00030 APPRO PICK,2
[00]PICK#00040 MOVE PICK
[00]PICK#00045 FINISH
```

CHAPTER 3 - BASIC MOTION AND LOCATION TEACHING COMMANDS

3-1 TEACHING A LOCATION

A location in RAPL-II is the final position of the arm where parts will be picked up, put down, machines controlled, fluids dispensed, and so on.

Normally, the robot arm is moved to the position you wish to save, and the location co-ordinates are stored in the user memory of the controller. Since most locations are stored by their cartesian coordinates, they can also be entered off-line.

How to move to a desired location:

There are several methods you can use to move the robot to a location you wish to save:

1. LIMP Mode
2. MANUAL Mode (Cylindrical or Joint)
3. JOINT command
4. JOG command

You have already tried the MANUAL and LIMP modes. These are the two most common ways to manipulate the robot for teaching. There are others however, such as the JOINT and JOG commands. These are incremental motion commands: JOINT works by moving one joint by a specified number of degrees, while JOG moves the robot Tool Centre Point by an incremental cartesian motion. The following sequence illustrates the use of these commands:

STEP 6) Set the programmed speed. Type >>SPEED 20<cr>.

STEP 1) Make sure arm is in READY position. Type >>READY<cr>.

STEP 2) Type >>JOINT <JOINT>:1 BY (DEGREES):-15<cr>.

STEP 3) Repeat this command for Joint 2 by -50, Joint 3 by -50, and Joint 4 by -80 degrees. The arm will be close to a place which can simulate picking up a part from the surface of the mounting plate (Make sure that the moves leave a couple of inches between the end of the gripper and the solid surface).

The gripper is now close-to but not exactly vertically aligned. To pick the imaginary part up, you will want it exactly vertical.

STEP 4) Type >>ALIGN <cr>, for the ALIGN command. The arm will move so that the centre of the gripper flange is in exactly the same spot, but the gripper is aligned vertically. This command will also align the tool axis horizontally if it is closer to horizontal when the command is executed. Note that this is the same as using the Manual Mode Align function switch.

1-7 ENTERING COMMANDS (Continued)

There are times however, when a mis-typed character cannot be rubbed out. RAPL-II interprets commands "on-the-fly". This means that as soon as an "element" of a command line has been completed, it has been entered and cannot be changed even though it may still be on a line which has not been completed with a <cr>. A command or argument is completed with either the space, comma or <cr> character.

⚠ **WHEN A MISTAKE HAS BEEN MADE IN A COMPLETED ELEMENT OF A COMMAND LINE, TERMINATING THE COMMAND LINE WITH A <CR> MAY RESULT IN AN UNEXPECTED MOTION. AN <ABORT> SEQUENCE MUST BE USED TO TERMINATE THE ERRONEOUS ENTRY.**

An **<ABORT>** stops whatever the robot is currently doing, including command line entry, program execution and any motion.

When the controller is first turned on, the RAPL-II syntax builder (HELP mode) is automatically activated. The syntax builder monitors your input as you type. As soon as you have entered enough characters to uniquely define the command you want, the syntax builder completes the command name and prompts you for any additional arguments are required.

Instructions which define commands to be entered list the letters required to make a unique command in **BOLD** and show the remainder of the complete command in normal text. For example:

>>**STA**TUS<cr>

Indicates that you must type the "STA" to generate a unique command. As soon as that has been entered, RAPL-II completes it by "typing" the "TUS". To enter the command, you type the <cr>. In this way, users who prefer to turn off the HELP mode can determine what the complete command is, and others know how many letters to type to make a command unique.

In many instances RAPL-II prompts you for more information with a message. In those cases, the first time the command is illustrated in this manual, the message will be shown. In subsequent examples of the command, it may be omitted.

If you do not wish to use the HELP Mode, issue the NOHELP or DISABLE HELP command to turn it off.

PREFACE

This manual provides basic user information for operating and programming the CRS Plus A100/200 Series Small Industrial Robot System. Additional information is available in the following documents:

* SRS-MAN/RAPL	RAPL-II Programming Manual
* SRS-MAN/1ATM	A100/200 Series Industrial Robot Technical Manual

2-1 **ARM POWER (Continued)**

Any commanded motion can be stopped by an **<ABORT>** sequence. An **<ABORT>** is especially useful if the robot is attempting to go somewhere that it cannot reach, such as into (or through) a stationary object. The Abort push-button on the Teach pendant is the most convenient way to stop motion during program testing etc. Note that the ABORT push-button can be used as a "Dead-man" switch: the **<ABORT>** is actually triggered when the button is released. In cases where you expect trouble or if a collision could cause a serious problem, you can hold the button in for as long as the machine continues to operate normally. However, as soon as you anticipate trouble, release the button, and the robot will immediately stop.

Should the robot actually collide with an object, the servo system detects the collision in approximately 0.5 seconds. When it does, RAPL-II issues an error which halts any commanded motion and limps the joint temporarily to release the voltage to the affected motors. If however the collision was caused by a failure in the servo system, perhaps because of a component failure, the robot may move at high velocity without computer control. Since this is not a commanded motion, an **<ABORT>** (which stops the command) will not stop it. In this case, the only way to stop the arm is to remove ARM POWER by pressing the large, red, button on the controller front panel. That is why the E-STOP button (as it will be referred to in the rest of this document) must be accessible at all times during robot operation.

STEP 2) Turn the ARM POWER off. Press the E-STOP button on the front panel.

Notice the arm once again becomes limp, or in the case of a brake-equipped model, the brakes click on and hold the arm still. Also notice that an error message appears on the terminal screen. The message should say "040 ARM POWER" and a new prompt has appeared. This error will have stopped any program execution and arm motion, and even interrupt any command entry you may have started. This is because any RAPL-II error has the same effect as issuing an **<ABORT>**.

Whenever the arm power is off, the controller, (having sensed that condition), puts the servos in a "follow" state. The position feedback continues to operate so the robot computer "knows" where the arm is, but it makes no attempt to control this position. This "follow" state is also called LIMP (see section 2-2 below). When the arm power is turned on again, the LIMP mode is de-activated and the controller again controls and corrects the arm position.

STEP 3) Turn the arm back on now. Again use the ARM POWER switch on the front panel. Note that the arm does not jump back to where it was but locks up in its current position.

3-3 GRIPPER COMMANDS (continued)

When in MANUAL mode, the toggle switch marked "GRIP" on the Teach pendant will open and close the servo gripper. In this case the "SPEED" dial on the Teach pendant indicates the percentage of full force the servo gripper is operating at.

⚠ SERVO GRIPPERS MUST NOT BE OPENED OR CLOSED AGAINST AN OBJECT WITH A FORCE OF MORE THAN 75% FOR MORE THAN 5 SECONDS. DOING SO COULD SHORTEN THE LIFE OF THE GRIPPER MOTOR.

Throughout the rest of this manual we will assume that a pneumatic gripper is being used. This convention is used for simplicity only. If a servo gripper is being used, it is expected that the reader will make a suitable command substitution.

5-2 BASIC INPUT/OUTPUT

Now go back to program PICK1. You just added a program line which changed the program flow. That is, if a certain logical operation had been satisfied (if the counter had reached five), the program will stop. To make the program more user-friendly, add some output statements to the program. After the following lines have been entered in the program, a message will be printed after each part has been moved, and when the program is finished.

STEP 1) Insert the following lines into your program:

```
120 STOP Finished FIVE Parts!
75 TYPE 'Part count is now: '
76 TYPEI COUNT
77 TYPE ''/
```

The TYPE command allows the string enclosed in quotation marks to be printed to the display screen. The TYPEI command will print a variable to the display screen in an integer format. This is important, because in RAPL-II, all variables are stored as real numbers (use the TYPEV command for real format output). Following a string in a TYPE command (even an empty one) with a "/" will cause a carriage return and line feed.

Run the program to see the result of these changes.

If you have a serial printer available, you could hook it up to the second RS232 port at the back of the controller (DEVICE 1). To have the data sent to the printer, you would replace the TYPE and TYPEI commands in the above lines with PRINT and PRINTI.

Now you can change the program to allow the operator to enter the number of parts to process. This requires the operator to input information to the controller. To do this, use the INPUT command.

STEP 2) Add the following lines to the PICK1 program:

```
7 TYPE 'ENTER NUMBER OF PARTS TO RUN'
8 INPUT MAX_NUM
117 TYPE 'FINISHED'
118 TYPEI MAX_NUM
119 TYPE ' PARTS'/
```

STEP 3) Now EDIT lines 112 and 120 in the PICK1 program to read:

```
112 IF COUNT EQ MAX_NUM THEN 117
120 STOP
```

Once again, run the program to see the result of these changes.

The program will start by asking you how many parts you wish to process. Respond by typing a number. The program will stop after that number of parts have been processed.

CHAPTER 6 - SUMMARY

Once you have completed the exercises in this manual, you will be ready to program your own application. This has been only a preliminary introduction to the functions of RAPL-II. All commands and features are documented completely in the RAPL-II Programming Manual and Robot System Technical Manual which are included with the robot user manual package. Chapter 3 of the RAPL-II Programming Manual gives a nice overview of the various commands grouped by function, while chapter 4 of the RAPL-II Programming Manual contains detailed descriptions of all the commands. APPENDIX A of the Robot System Technical Manual contains a complete list of all commands protected by the RAPL-II password.

CRS Plus or your CRS distributor can also provide application notes and assistance with programming questions.

Training programs on programming, using, maintenance, and service of CRS robot systems are available from CRS. These courses are informative and highly recommended for all users of CRS robot systems. A list of courses is included in APPENDIX A of this manual.

4-4 CREATING A "LOOPING" TASK

What if you want this program repeated? The RUN command can use an argument to indicate a number of repetitions.

STEP 1) Type >>**RUN ,2<cr>**. This is equivalent to "RUN PICK,2" since RAPL-II defaults again to the "current program executing". The comma tells RAPL-II that the "2" is a second argument and not part of the program name.

This command will cause the PICK program to be run through twice as you will see.

STEP 2) The program can be run indefinitely by entering >>**RUN PICK,F<cr>**. The "F" stands for Forever.

If you have followed these instructions exactly, it should now be running this program indefinitely at a very slow speed. You have checked out the program logic and it seems to work alright. Now you must get the system operating at a more reasonable speed.

STEP 3) To do this you must first stop the program. You could issue an <ABORT> to stop, but this is a bit drastic. Try typing <Ctrl-A>. This terminates execution of the program at the end of the next instruction. It is not so much of a panic stop as the other two methods; it is much gentler.

STEP 4) When the prompt returns (after execution of the step that RAPL-II is decoding) set the speed to 100 with the SPEED command. Type >>**SPEED 100<cr>**.

STEP 5) Since the robot stopped neatly at the end of a line of code, you can use the PROCEED command to continue the program from where it stopped. without having to restart the program at the beginning. Type >>**PROCEED <cr>**.

[However, if the next step happens to be the STOP or another program branching command, it may not proceed properly and will just return a prompt. If this happens, re-start the program with the RUN command.]

STEP 6) Stop the program now using a <Ctrl-A>.

Another way to have the program run indefinitely is to use the GOTO command to cause program flow to jump to some previous line. In this case, a logical place to return to would be line 30.

STEP 7) To do this type: >>**115 GOTO 30<cr>**.

List the program to see that it has put all of these lines in the correct location.

STEP 8) Type >>LISTP<cr>.

The program listed will be the one currently being edited unless a program name is given (ie. LISTP PICK<cr>). You should see this on the screen:

1-6 POWER-ON PROCEDURE (Continued)

If no characters at all appear on the screen, check for loose connectors in the connection between the terminal and the robot controller, turn the controller main power off, and try again.

If some characters appear but they are garbled, check your terminal communication parameters. Baud rate should be 9600, no parity, 8 data bits, 1 stop bit and no echo. If you are using your own terminal, it should have documentation on how to set these parameters up.

ROBCOMM

When running Robcomm-II in Terminal Emulation mode, the communication parameters are correctly set up by the software. The terminal communication baud rate is programmed in the Set-up menu. Initially it should be set to 9600 Baud.

If the terminal is correctly set up and the message is still garbled, perform a Teach-Start (Press and hold the Teach button on the Teach Pendant while turning the MAIN power switch to ON). The Teach-Start will reset the communication parameters in the robot controller.

Following the sign-on message is a "prompt". The normal RAPL-II prompt is a double-angle bracket ">>". The prompt indicates that the controller is ready to accept a command from the terminal. It is also used to tell you what mode of operation the robot is in. The double angle prompt indicates the robot is in **Immediate** mode. This means commands entered from the terminal are executed immediately after a carriage return (<cr> or <enter>) is entered.

NOTE: Do not turn on the arm power at this time (ie. do not press the ARM POWER switch).

1-7 ENTERING COMMANDS

Before attempting to enter commands in RAPL-II, you should be aware of some special considerations concerning the way RAPL-II deals with correcting typing mistakes. If you enter a character by mistake it can usually be erased by using the <del> key. This key must issue an ASCII "7F" (hexadecimal) character. Consult your terminal manual for the key which issues this character.

ROBCOMM

When running in Robcomm-II Terminal Emulation mode the <Backspace> key issues the correct "rub-out" instruction to RAPL-II.

CHAPTER 1 - GETTING STARTED
1-1 INTRODUCTION 1-1
1-2 SOME TERMINOLOGY FOUND IN THIS MANUAL 1-1
1-3 REQUIRED EQUIPMENT 1-2
1-4 ROBOT MOUNTING 1-2
1-5 HOOK-UP ALL COMPONENTS 1-3
1-6 POWER-ON PROCEDURE 1-3
1-7 ENTERING COMMANDS 1-5
1-8 SYSTEM CHECK OUT 1-7

CHAPTER 2 - STARTING UP THE ROBOT
2-1 ARM POWER 2-1
2-2 LIMP MODE 2-3
2-3 HOMING THE ROBOT 2-3
2-4 MANUAL MODE 2-6

CHAPTER 3 - BASIC MOTION AND LOCATION TEACHING COMMANDS
3-1 TEACHING A LOCATION 3-1
3-2 BASIC MOTION COMMANDS 3-3
3-3 GRIPPER COMMANDS 3-4

CHAPTER 4 - BASIC PROGRAMMING TECHNIQUES
4-1 USING THE EDITOR 4-1
4-2 RUNNING A PROGRAM 4-3
4-3 PROGRAM LINE ENTRY 4-4
4-4 CREATING A "LOOPING" TASK 4-5
4-5 TRACE MODE 4-6

CHAPTER 5 - MORE ADVANCED PROGRAMMING TECHNIQUES
5-1 COUNTERS AND VARIABLES 5-1
5-2 BASIC INPUT/OUTPUT 5-3
5-3 DIGITAL I/O 5-5
5-4 SUBROUTINES 5-7

CHAPTER 6 - SUMMARY

2-2 LIMP MODE

????

In the section above, you saw that the controller put the servos into LIMP when the arm power is turned off. It is possible to command the servos to be LIMP with the arm power turned on as well. In this state, the arm is free to drop due to gravity, to be pushed around by the operator or move due to any other externally applied force. As you will see, this can be a convenient mode for programming the system.

STEP 1) To enter LIMP mode, type >>**LIMP** AXIS # OR CR :**<cr>** for the command. Be prepared to catch the arm if it drops! A message on the screen informs you to type "NOLIMP TO END" to leave LIMP mode.

When you grasp the end of the arm in the LIMP mode, you can push the arm around into any desired position. This is an effective way of moving the arm quickly to a desired location.

STEP 2) One axis at a time can be LIMPed by adding the axis number argument to the LIMP or NOLIMP commands. Try for instance typing

```
>>LIMP 1<cr>
```

You will find that the waist is limp but all other axes remain locked.

STEP 3) One axis can be locked. Limp the complete arm again and type:

```
>>LIMP <cr>
>>NOLIMP AXIS # OR CR :2<cr>
```

Only joint 2 will be locked.

STEP 4) Before proceeding, leave the LIMP mode by typing:

```
>>NOLIMP <cr>
```

2-3 HOMING THE ROBOT

When the controller is first turned on, it does not know the position of the arm relative to its world (which in this case is the arm's mounting surface). In order to perform pre-programmed tasks, the controller must somehow be informed of this vital information. The sequence which does this is called "HOMING THE ROBOT".

A precise location has been stored in the controller RAM memory at which the controller and the arm can be synchronized. The Homing sequence involves moving the arm close to this location then instructing the robot to move each axis the last few degrees to reach this stored location precisely. The READY position (see FIGURE 2-1 below) is used as the starting point for the HOME sequence when the robot is shipped from the factory. You can change this location to suit your application later.

5-4 SUBROUTINES (continued)

The above two programs are useful when there are several locations to pick from, but only one location to place the parts at. The first program (MAIN) is the "executive" program, and the one that would be executed with the RUN command. All that MAIN does is to set up the locations for the pick and place cycle, while the call to the second program (PK_N_PLC) does the actual motion.

The GOSUB command in the first program instructs the controller to start executing the second program. The names and numbers after the subroutine name in the GOSUB command are parameters that are passed on to the called program. The RETURN command in the second program returns control to the calling program.

Using this method, every location must be stored in memory rather than just one location which is changed using the SHIFT command as in program PALLET. This approach uses more memory which could be a problem for large pallets.

Notice in the subroutine the use of %0, %1 and %2. These are special variable names that may only be used in called subroutines. These variable names are assigned the values of the parameters from the calling GOSUB command. Thus, in line 110 GOSUB PK_N_PLC PICK,2,PLACE, %0 is assigned the location PICK, while %1 is the constant 2 and %2 is assigned the location PLACE. Up to eight parameters may be passed from the calling program to the subroutine using this method (%0 to %7).

Interrupting execution of a program while in the subroutine will leave the subroutine as the default "currently executing program" rather than the calling program. Take this into account when using the RUN command without an argument. Trying to RUN a subroutine will cause an error. The RETURN command looks for a destination line number from the calling routine to return control to, and of course will not find one.

4-2 RUNNING A PROGRAM (Continued)

STEP 4) Type >>**RUN PICK<cr>** to run the program. The robot should move from POINT_1 to POINT_2 as planned.

After the sequence is complete, the first thing you will notice when you turn your attention back to the display screen is an error message. This is because there was no proper termination of the program. A STOP command is needed.

If you were paying close attention to the robot as it moved (and you should have been) you would have seen that the gripper did not open and close at POINT_1 or at POINT_2. This is because of the way RAPL-II deals with command sequencing. After it sets up any robot move command, it starts the motion and then goes on to the next command. In this case, it executes the OPEN or CLOSE commands very quickly after processing setting up the next motion commands.

This strategy speeds up program execution, but at times it can lead to problems in synchronizing. To overcome this, you must use the FINISH command. This command will cause the robot to hold execution of the next step of the program until the current motion command is completed.

4-3 PROGRAM LINE ENTRY

Program Line Entry (PLE) mode allows you to insert commands into the current program being edited without having to use the editor. To enter PLE mode, enter a number at the **beginning** of a command line, when in Immediate or Manual mode. The line will automatically be inserted into the program currently being edited (see chapter 1-6 of the RAPL-II Programming Manual).

STEP 1) To illustrate, insert FINISH commands just before the OPEN and CLOSE command lines of the program PICK. To make these changes, type:

```
>>45 FINISH<cr>
>>85 FINISH<cr>
```

STEP 2) Now let's add the STOP command to the end of the program. Type:

```
>>120 STOP<cr>
```

RAPL-II will insert these lines in numerical sequence in the correct place. Try running the program again and see the differences these changes make.

STEP 3) This time you just have to type >>**RUN<cr>**. A RUN command with no argument causes RAPL-II to run the program which was last executed. This "current program executing" can be checked by a STATUS command.

STEP 2) To remove the pencil from the gripper, issue the open command. If using a Servo Gripper, enter >>OPEN GRIPPER OPENING FORCE(%): **60<cr>**. If you are using an air gripper, enter >>OPEN <cr>.

CYLINDRICAL Manual Mode

Since the robot has already been Homed, you can use the CYLINDRICAL Manual mode and use some of the additional features of the RAPL-II MANUAL Mode.

In Cylindrical Manual mode, joints two and three are synchronized so that they will move in a purely radial (towards or away from the base) or in a vertical fashion (parallel to the Z-axis). The toggle switch that controls joint 2 in Joint Manual mode, controls the radial movement in Cylindrical mode. Toggle switch three which controls joint three in Joint Manual mode, controls the vertical movement in cylindrical mode.

Try moving the arm about in CYLindrical MANUAL mode. Repeat the pencil-picking exercise using the new mode of operation.

STEP 1) At the prompt, type >>**MA**NUAL JOINT OR CYLINDRICAL MODE? **CYL<cr>** for CYLindrical Manual mode. Note that the prompt is now C>.

STEP 2) Grasp the arm by the gripper with one hand while holding the Teach Pendant in the other. Flip toggle switch 8 over to the side labelled "LIMP". The arm will go limp. Position the arm so the gripper fingers are just above the pencil and flip switch 8 to the "NOLIMP" side.

STEP 3) The robot is now roughly positioned to grasp the pencil. To line up the gripper to get the pencil, press the toggle switch labelled "ALIGN" (toggle switch 7) to either side. Notice the gripper should now be pointing straight down.

STEP 4) Now use switches 1 and 2 (angular and radial motion) to position the fingers exactly above the pick up point. As with JOINT mode, it is best to start very slowly and increase speed after pressing the switch. Once there, use switch 3 to move down to the precise point desired and again close the gripper.

Once you have become more familiar with the manual mode, you will find the LIMP, ALIGN and CYLindrical features a great aid in positioning the arm. Even in comparing the ease of moving the robot to grasp the pencil, you may have noticed how much faster this technique is compared to positioning using JOInt manual mode only.

Turn off the Manual mode using the NOMANUAL command.

STEP 5) Type >>**NOMANUAL<cr>** to return control to the Immediate mode. Note that the screen prompt returns to '>>' (the Immediate mode prompt).

1-6 POWER-ON PROCEDURE (Continued)

STEP 2) The robot system Main Power Switch may now be turned on. Check the display screen for the RAPL-II sign-on message.

```
RAPL-II ROBOT AND AUTOMATION PROGRAMMING LANGUAGE
COPYRIGHT CRS PLUS
VERSION 1.02  FEB 20, 1990
A150 ROBOT              (This line may change depending on your robot type)
```

The normal RAPL-II sign-on message

If no message appears:

RAPL-II checks the status of the hardware installed in the controller before issuing the sign-on message. There are three messages which can occur before seeing the normal sign-on message:

EXTRA CARD DETECTED, ENTER PROPER NUMBER OF AXES:
- in this case, RAPL-II has detected at least one more axis card than had been installed the last time the system was run. Enter the number of axis cards currently in the system.

CARD MISSING, ENTER PROPER NUMBER OF AXES:
- in this case, RAPL-II has detected at least one less axis card than had been installed the last time the system was run. Enter the number of axis cards currently in the system.

GRIPPER CARD NEEDED? (Y/N) : Y "A"IR, "S"ERVO, "M"AGNETIC:
- in this case, RAPL-II has detected a card in slot 8 of them mother board which was not installed the last time the system was run. AS SOON AS THE "Y" IS ENTERED, THE PROMPT FOR GRIPPER TYPE COMES UP. One of the optional type designations (A, S, or M) must be entered before the <cr> is typed

In addition, RAPL-II checks the user memory to ensure that it is intact. If it has been scrambled, the following message will be displayed:

```
*** USER MEMORY CHECK FAILED ***
ALLOCATION WILL ERASE USER MEMORY. CONTINUE? (Y/N)
```

In this case, there is usually no option but to answer yes to the question and proceed to allocate memory. Refer to the RAPL-II Programming Manual for details on this procedure.

If any of these messages appear, they must be answered satisfactorily before the controller will permit you to proceed.

CHAPTER 1 - GETTING STARTED

1-1 INTRODUCTION

The CRS Plus Small Robot System is a complex piece of equipment. We recommend that anyone who will use the robot system should participate in a CRS Plus introductory course on the use of this machine. We also highly recommends the advanced programming course to any user who will be programming the system. Contact CRS Plus or your distributor for details.

This manual is intended to help the robot programmer gain an understanding of the operation of the CRS Plus robot system. To accomplish this, you will be working through some basic commands, teaching robot locations, and writing simple programs to teach the robot tasks. A more thorough description of RAPL-II, (the language of the robot system) is contained in the RAPL-II Programming Manual. Installation and maintenance details are described in the Technical Manual. This manual will refer to these other documents so you should have access to these documents as you work through it.

1-2 SOME TERMINOLOGY FOUND IN THIS MANUAL

<cr>	Indicates the "ENTER" key found on most terminal or PC keyboards. It is referred to as RETURN on some keyboards.
<Ctrl-?>	Where the ? is a letter. This is the so-called Control-Character set. It is entered by holding down the Control key (often labelled "Ctrl") on the keyboard while pressing the letter.
<ABORT>	A RAPL-II abort sequence. This is either <Ctrl-C> or a <Ctrl-X> from the keyboard or the ABORT push-button on the Teach Pendant.

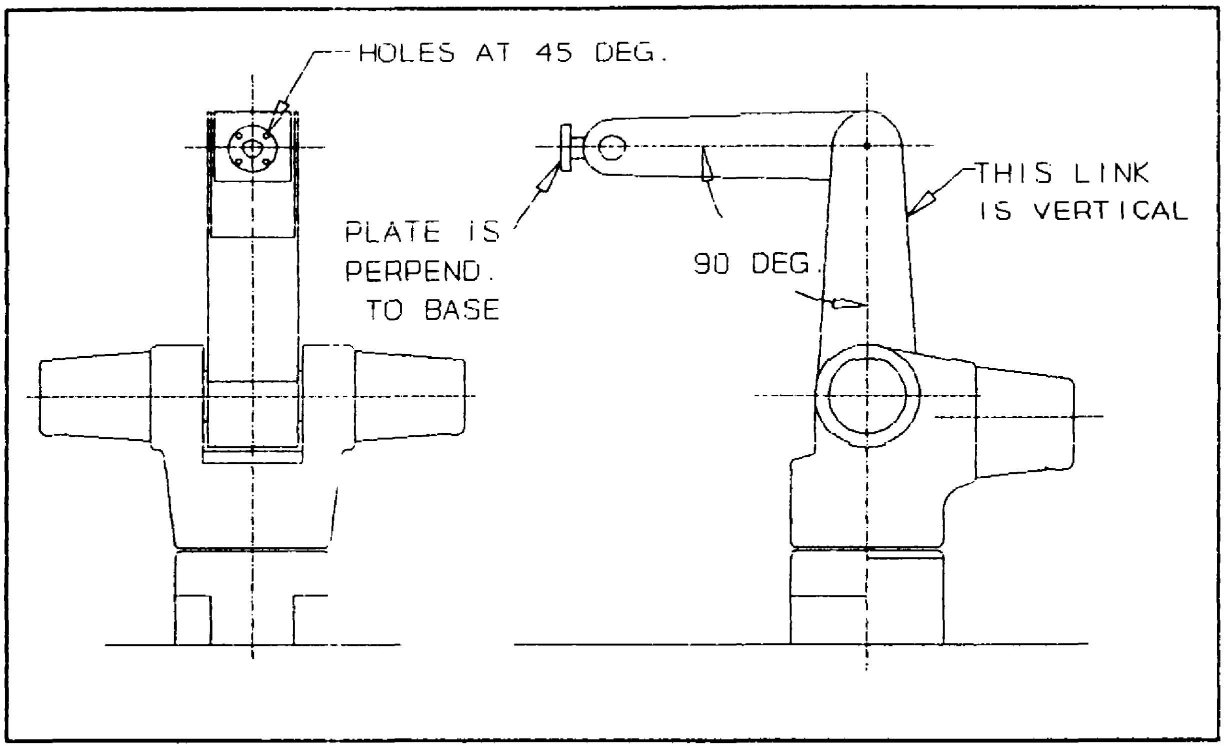

FIGURE 2-1 THE READY POSITION

STEP 1) Position the arm at roughly the READY position by lining up the HOMING marks on the arm (See FIGURE 2-2). The arrow on the moving part of the joint must point to somewhere in the range of the mark opposite to it as in the figure. The arm can be moved to this approximate location in the LIMP mode or with the arm power turned off. If you use the LIMP Mode, be sure to issue the "NOLIMP" command before proceeding. If you move there with the arm power off, turn it ARM POWER back on as soon as the marks line up.

CHAPTER 4 - BASIC PROGRAMMING TECHNIQUES

4-1 USING THE EDITOR

Programs are stored in the controller RAM memory. You have already done a DIR command to see which programs are stored there. The next section shows how to enter a program.

At any time, only one program can be edited even though several can exist in memory. The program being edited is changed using the EDIT command.

STEP 1) Type >>EDIT PICK<cr>. This will start editing a program called "PICK". Since no "PICK" program exists in memory, the controller creates one. You can of course name the program anything you like, but it makes sense to call it something that reminds you of its function.

You have now entered the Edit mode. The prompt is an asterisk like ">*" instead of the former ">>".

STEP 2) For now exit this mode by typing >*END. The Edit mode command END puts you back to the immediate mode. The ">>" prompt will return.

STEP 3) Check that you have created a program called "PICK" by the using DIR (type >>DIR<cr>) command. Note that the size of the program is 00000 because you have not added anything to it yet.

STEP 4) Use the STATUS command to show that the program currently being edited is "PICK".

STEP 5) To enter the edit mode again, use the EDIT command without a program name. The screen will return the following:

```
>>EDIT  <cr>

EDITING  PICK

>*_
```

You are now ready to enter a program in Edit mode. Take care to type the following lines in exactly as shown. The editor performs no syntax check on a line before it is entered into memory. Thus any errors made now will not be detected until the program is actually run.

STEP 6) Type I, for Insert. The Editor will complete the command and your line will look like this:

```
>*INSERT
```

5-3 DIGITAL I/O

Many (if not all) applications require sensors or switches detecting conditions in the work space and/or the ability to turn on an external device. This type of interaction is known as "digital" or sometimes "switch-closure" I/O because it has only two states: on or off.

Typical uses of digital inputs are:
- Part detection,
- Safety switches in the work place,
- Machine ready switches,
- Part-type selector switches,
- Air-cylinder end of stroke switches,
- or Part-in-gripper switches.

Typical uses of digital outputs are:
- Air solenoid valves for cylinders,
- Operator warning lights,
- or Machine cycle-start signals.

All CRS robot systems support these types of functions through an array of I/O points from the master processor. These computer-level signals are protected from the industrial environment by a buffering scheme using one or more of the optional DB-16/R I/O expansion racks. Buffering relays can be supplied for any of the common industrial voltage levels.

RAPL-II uses several commands to access and control the digital I/O. A complete list can be found in the RAPL-II Programming Manual Section 3-3. For this manual, you will be introduced to just 3 commands: OUTPUT, WAIT, and IFSIG.

The OUTPUT Command

This command controls the state of all digital outputs. It uses a number to indicate the output to be set (positive number) or reset (negative number).

example:
```
550 OUTPUT 1
1010 OUTPUT -16
2000 OUTPUT 1,2,-4,6,-8
```

The WAIT Command

This command halts program execution until a certain input state is satisfied. Again it uses a number to indicate the input state to be waited for: a positive number means wait for that number to be on, a negative number means wait for that number to go off.

example:
```
100 WAIT 4
200 WAIT -6
```

5-4 SUBROUTINES

If there is a section of code that is executed several times in the same routine, you may wish to use a subroutine to simplify the writing and logic of the program. A subroutine is called from a main routine (or even from another subroutine) using the GOSUB command.

To illustrate the use of subroutines look at the following programs:

```
PROGRAM MAIN:
10 ;
20 ; Initialize the program:
100 SET PICK = PICK_1
110 GOSUB PK_N_PLC PICK,2,PLACE
120 SET PICK = PICK_2
130 GOSUB PK_N_PLC PICK,2,PLACE
140 SET PICK = PICK_3
150 GOSUB PK_N_PLC PICK,2,PLACE
160 SET PICK = PICK_4
170 GOSUB PK_N_PLC PICK,2,PLACE
180 SET PICK = PICK_5
190 GOSUB PK_N_PLC PICK,2,PLACE
200 ;
210 STOP All finished
$

PROGRAM PK_N_PLC:
100 ;
110 ; Routine to pick and place part.
120 ;
200 APPRO %0,%1
210 MOVE %0
220 FINISH
230 CLOSE
240 DEPART %1
250 APPRO %2,%1
260 MOVE %2
270 FINISH
280 OPEN
290 DEPART %1
300 ;
310 ; Return to main routine:
320 ;
400 RETURN
$
```

4-1 USING THE EDITOR (Continued)

STEP 14) In the last line you made an error (we asked you to!). You wanted line number 110, not line number 105. To correct this, use the Move command.

```
>*MOVE105,110<CR>
```

STEP 15) Once again, end the editing session. Use the END command. This will bring you back to Immediate mode (denoted by the prompt changing to '>>').

4-2 RUNNING A PROGRAM

Before attempting to run the program, list it and compare it to the one below. If they are different, then you should go back into Edit mode and correct it. This is done by entering the EDIT mode, deleting the incorrect line, and re-entering it.

STEP 1) Type >>**LISTP PICK<cr>**, for a listing of the program. The screen will display the following:

```
PROGRAM: PICK
10 OPEN
20 READY
30 APPRO POINT_1,2
40 MOVE POINT_1
50 CLOSE
60 DEPART 2
70 APPRO POINT_2,2
80 MOVE POINT_2
90 OPEN
100 DEPART 2
110 READY
$

>>_
```

The dollar sign at the end of the list of program lines is a symbol used by RAPL-II to determine the end of the program.

This program is a simple pick-and-place routine to take a part from POINT_1 and move it to POINT_2. Try running the program now. When trying a program out for the first time, always use a slow speed and be ready to abort the program should anything unusual happen.

STEP 2) Set the speed to 30% with the SPEED command. Type >>**SPEED 30<cr>**.

STEP 3) When first running a new program, it is a good idea to hold the Teach pendant in one hand with a finger over the ABORT button. If the robot does anything surprising, you can then press and release this button. The <ABORT> sequence stops the robot immediately and halts program execution.

2-4 MANUAL MODE

JOINT Manual Mode

Next, try out the Manual mode of moving the robot. Before proceeding, make sure that the Teach pendant is plugged into the front panel receptacle.

STEP 1) Type >>**MANUAL** JOINT OR CYLINDRICAL MODE? JOI<cr> for the command. Notice that the RAPL-II prompt has changed to "J>". The "J" indicates the Joint Manual mode. Joint MANUAL mode is the only mode available until the robot has been Homed.

STEP 2) Holding the Teach pendant, examine the labels on the eight switches down the right-hand side of the pendant. The first five are labelled with the names of the five axes of the robot. The bottom three are numbered 6 through 8, and list some special commands. For now consider the first five switches only.

STEP 3) Pick up the Teach Pendant and try some simple moves. Rotate the speed knob counter-clockwise until it stops (i.e.- set speed to zero). Push the first switch to the right or left and slowly increase the speed (rotate the speed knob clockwise) until the arm moves. Slow down and stop by using the speed knob and the direction switch. Try each axis in turn.

ALWAYS START MANUAL MOTION AT A SLOW SPEED TO AVOID COLLISIONS.

STEP 4) Use the pendant switches to move the robot arm to pick up a pen or pencil from the table surface. Once at the final destination, manipulate the gripper using the 'GRIP" toggle switch on the Teach pendant. If you have a servo gripper, the "SPEED" dial is used to control the gripping force. Set it to about 50 and use the "GRIP" toggle switch to close the gripper.

Leaving MANUAL Mode

There are two methods to leave manual mode:

1) A direct command. The command is NOMANUAL.

2) Any motion command in which the arm is controlled by the computer only (rather than the Teach Pendant), causes the manual mode to be disabled.

Use the latter in this case:

STEP 1) To leave MANUAL this time, type J>**READY** <cr>. Note that the prompt returns to the normal >>.

1-5 HOOK-UP ALL COMPONENTS

STEP 1) Connect the AC power cord for the robot controller between the rear of the controller and a 3-prong wall socket.

STEP 2) A serial Video Display Terminal device (VDT) must then be connected to the controller with a cable using instructions found in Chapter 3 of the Technical manual. Connect the VDT to serial DEVICE 0 on the rear panel.

ROBCOMM

For installation instructions for the ROBCOMM-II hardware, refer to the ROBCOMM-II manual.

STEP 3) The robot arm must be connected to the controller. The large, grey, "umbilical" cables plug into the connector at the base of the arm and the connectors labelled "ROBOT ARM" on the controller. These cables are keyed to prevent mis-connection.

STEP 4) If one (or more) I/O Racks will be used, connect it (or them) to the controller as well. Each module is supplied with a flat ribbon cable that should be plugged into the lowest numbered available "DIGITAL I/O" connector on the back of the controller.

STEP 5) Finally, connect the Teach Pendant to the connector on the front panel of the control box.

The robot system is ready for use.

1-6 POWER-ON PROCEDURE

STEP 1) Turn on the video terminal device or PC using Robcomm-II first.

ROBCOMM

Enter terminal emulation mode in Robcomm-II. To do this, use the Alt-T key or the Terminal selection on the opening menu. Follow any instructions given on the screen. Refer to Robcomm-II Manual for more detail.

1-3 REQUIRED EQUIPMENT

Before starting this tutorial you will need a minimum of the following:

- A100/200 Robot arm
- A100/200 Robot Controller
- Teach Pendant
- Air or Servo Gripper
- Gripper Fingers
- The complete Robot Manual set.
- Serial Terminal _or_ IBM-PC (or fully compatible) computer with the ROBCOMM package.

ROBCOMM

All references to actions at the terminal will be accompanied by instructions for use with ROBCOMM wherever there is a difference. These will be contained in a block like this one.

1-4 ROBOT MOUNTING

The robot must be firmly fastened to a sturdy table during all operations. Otherwise, inertia from fast motions may cause the robot to "walk" around or tip onto its side. Further mounting details can be found in chapter 3 of the Technical Manual.

A suitable location must also be found for the controller, video terminal, and other options such as the digital I/O module, Operator Panel, Expansion Amplifier etc. where used. The controller, I/O Module, Operator Panel and Expansion Amplifier are designed for mounting in a standard 19" rack. All equipment must be positioned so you can easily reach the arm and the emergency stop which is the large red "mushroom" button on the front panel of the controller.

STEP 2) Once all marks have been lined up as shown as "STARTING FORM" in Figure 2-2, type:

>>**HOME** LOCATED IN ITS HOME BOUNDS? **Y<cr>**

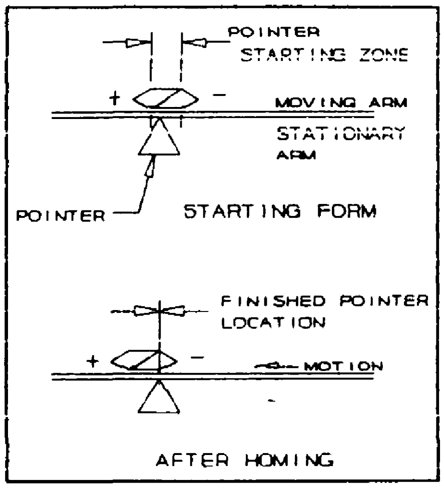

FIGURE 2-2 Use of the Homing Marks.

Now the controller takes over and accurately moves each joint toward the Home point. Notice that the joint moves at a very slow rate (5% of full speed) until the arrow points at the end of the opposite Homing mark (as shown as "AFTER HOMING" in Figure 2-2). As each joint reaches its Home location exactly, the screen displays a message that the HOME test has passed for that joint. If a test of any axis fails, or the arrow does not line up properly when the test has passed, you must repeat the entire Homing procedure. Note that the Home position may look "awkward" and seemingly random, but when the Homing sequence is completed successfully, the robot controller knows where the arm is.

Move the robot back to the READY position (the point from which a HOME command is executed) by doing the following:

STEP 3) Set the robot speed - A good speed to start with is 40% of full speed. Type >>**SPEED** (% OF FULL) : **40<cr>**. Use the STATUS command to verify that the motion speed is now 40% of full.

STEP 4) Move the robot to the READY position - Type >>**READY<cr>**. The robot will move at 40% speed to an upright position. Check this position carefully against Figure 2-1. If it seems out of alignment, repeat the HOME Sequence.

STEP 7) Without adding a space after the Insert command, type 10 OPEN. Your line will now look like this:

```
>*INSERT10 OPEN<cr>
```

STEP 8) Attempting to Insert a line with a space between the line number and the Insert command will result in an error.

```
>*INSERT 20 READY<cr>
>*028-Bad Line Number
```

STEP 9) Now let's enter the line properly.

```
>*INSERT20 READY<cr>
```

STEP 10) To take a look at a program line, use the editor List command. RAPL-II's editor will display an entered line one at a time. The procedure will appear as such on the screen:

```
>*LIST
10<cr> OPEN

>*_
```

STEP 11) Suppose that our next line number 30, was mistakenly entered as line number 20:

```
>*INSERT20 APPRO POINT_1,2<cr>
```

This will give us an error:

```
>*028-Line Exist
```

STEP 12) Enter the next six lines properly, as was done in step 7:

```
>*INSERT30 APPRO POINT_1,2<cr>
>*INSERT40 MOVE POINT_1<cr>
>*INSERT50 CLOSE<cr>
>*INSERT60 DEPART 2<cr>
>*INSERT70 APPRO POINT_2,2<cr>
>*INSERT80 MOVE POINT_2<cr>
```

STEP 13) The next three lines of the program, are duplicates of previous lines. Rather than re-typing each one, copy them from the originals. Type:

```
>*COPY10,90<CR>
>*COPY60,100<CR>
>*COPY20,105<CR>
```

The IFSIG Command

This command branches depending on the state of all digital inputs. Again it uses a number to indicate the input state to be checked: a positive number means check for that input to be on, a negative number means check for that input to be off. The program will branch if all states match the list in the command line.

example

```
100 IFSIG -4 THEN 160
110 IFSIG 4,-10 THEN 210
```

The correct operation of these commands depends on the configuration of your system. This manual cannot assume you have a DB-16/R with a certain number of accessible switches and output devices. We can simulate the operation of the digital input commands using the ability of RAPL-II to access the AUTO START switch on the front panel. The ONSTART command works like the WAIT command and the IFSTART command works like the IFSIG command.