A100/200 Series
SMALL INDUSTRIAL ROBOT SYSTEM

TECHNICAL MANUAL

CRS Plus Inc., 330 Harrington Ct. Burlington, Ontario, L7N 3N4
CANADA

REV.	REVISION HISTORY	DATE
-001	Original Issue	12/85
-002	Combine Installation & Technical	2/86
-003	Add additional information	5/86
-004	Version 3.30 Support for data base pointers in ACI	9/86
-005	Improved support for ACI functions Valid for RAPL version 3.46 or later	2/87
-006	Technical references in Appendix F Support for SRS-M1A	6/87
-007	RAPL Version 4.10 Support	2/88
-008	RAPL-II and A100/200 Series support	7/90

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

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

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

This manual provides general information, installation instructions, information, principles of operation for the CRS Plus A100/200 Series Small Industrial Robot System. Additional information is available in the following documents:

* RAPL-II Programming Manual

* SRS Series Small Industrial Robot System - Tutorial Manual

* SRS Series Small Industrial Robot System - ROBCOMM Software Manual

CONTENTS

CONTENTS (Continued)

FIGURES

FIGURES (Continued)

TABLES

1-1 SYSTEM DESCRIPTION

The A100/200 Series Small Industrial Robot System is a self-contained, five (5) axis, D.C. servo driven robot. It consists of an articulated robot arm, and the RSC Robot System Controller. Four models are available: the system can be supplied with two types of controller and two types of arm. The controllers are the RSC-PID8 or RSC-P8 controllers, the two arm configurations are with or without brakes.

DESCRIPTION	MODEL
RSC-P8 Controller/Arm without brakes	A150
RSC-P8 Controller/Arm with brakes	A151
RSC-PID8 Controller/Arm without brakes	A250
RSC-PID8 Controller/Arm with brakes	A251

The A100/200 Series Small Industrial Robot System is a complete free-standing robot system. However, it requires a Video Display Terminal (VDT) - (serial I/O) or an IBM PC or compatible for running the ROBCOMM communication package.

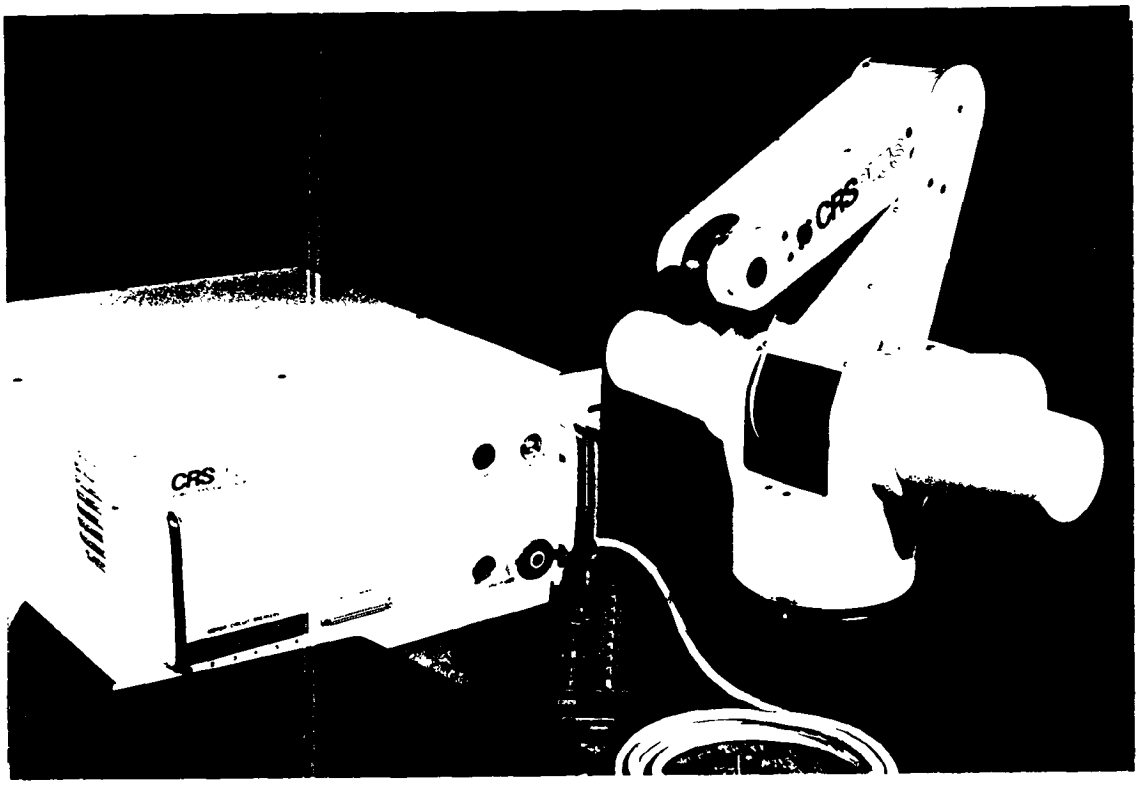


FIGURE 1-1 A150 Series Small Industrial Robot System

1-1 SYSTEM DESCRIPTION (Continued)

The A100/200 Series also requires a gripper or tool to perform its intended task. CRS offers several optional end-effectors for specific tasks:

SGRIP - An electric Servo controlled parallel motion 2-finger gripper with controller card. Finger travel is 0-2" with programmable position and force. It is also capable of measuring objects between its fingers.

PGR112/3 - A 2-jaw double acting air gripper with 3" long angular motion re-toolable fingers. Travel is 0-10 degrees per finger.

MGRIP - A small magnetic gripper with controller card intended mainly for experimental and educational use. Programmable magnetic strength.

1-2 ROBOT ARM

The A100/200 Series articulated arm consists of five major components. The base, shoulder, upper arm, lower arm and wrist. The arm has five degrees of freedom (joints): the waist, shoulder, elbow, wrist roll and wrist pitch.

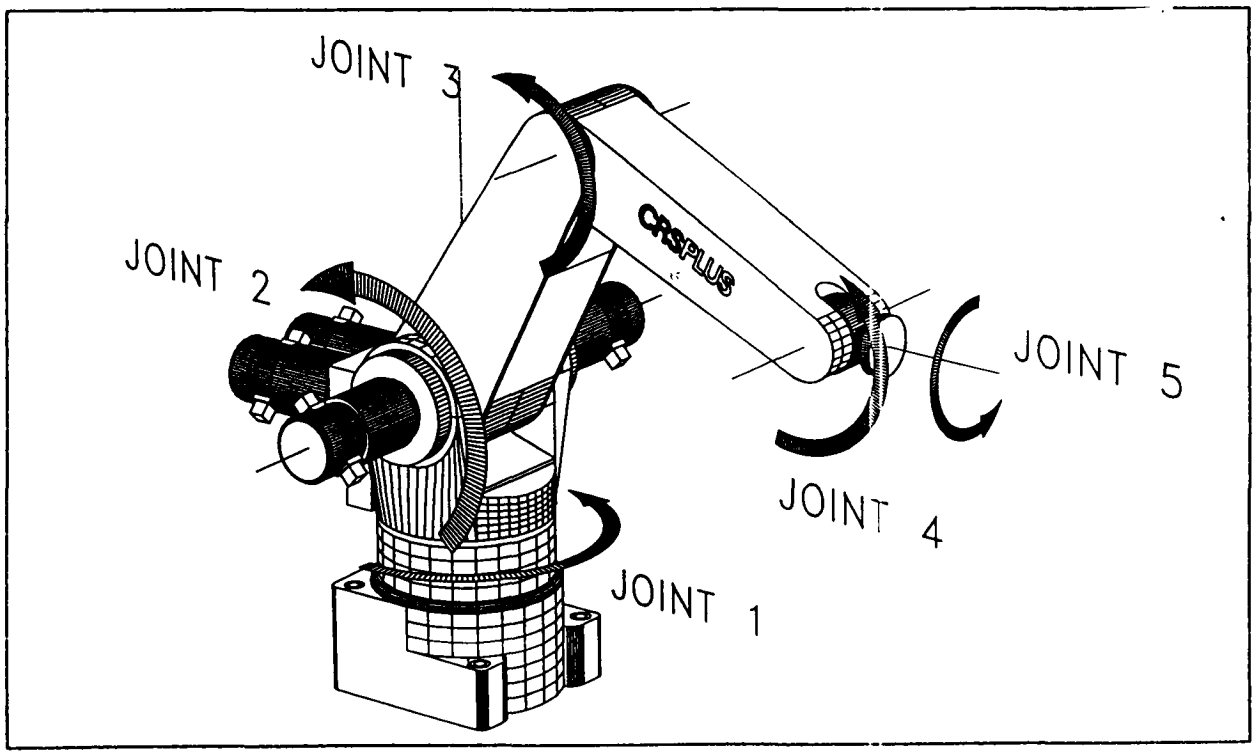


FIGURE 1-2 A100/200 Series Robot Arm

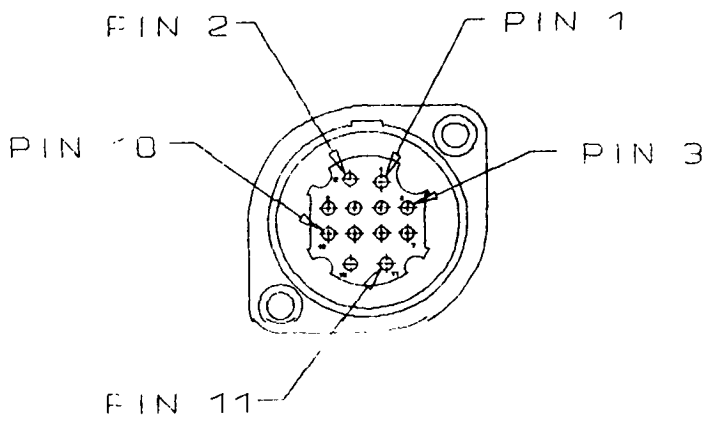
1-2 ROBOT ARM (Continued)

Arm Brake System

There are two types of arms in the 100/200 Series: the 150/250 model and the 151/251 model. The latter systems include fail-safe electro-magnetic brakes on joints 2 through 5. The brakes are energized whenever the system ARM POWER is on. When the arm power is turned off, power to the brakes is interrupted and the arm is held in position.

The brake circuitry is protected by a fuse on the encoder connector board (see Section 7-10). This fuse is rated at 2 Amps, 250 volts slow-blow.

The supply from the controller to the arm for the brake power is through the encoder signal cable. All brakes are energized with the same signal. Each brake has its own return path however: through the motor return line. This simplifies the wiring to the motor.



Connector type: Hiroshi RP-17

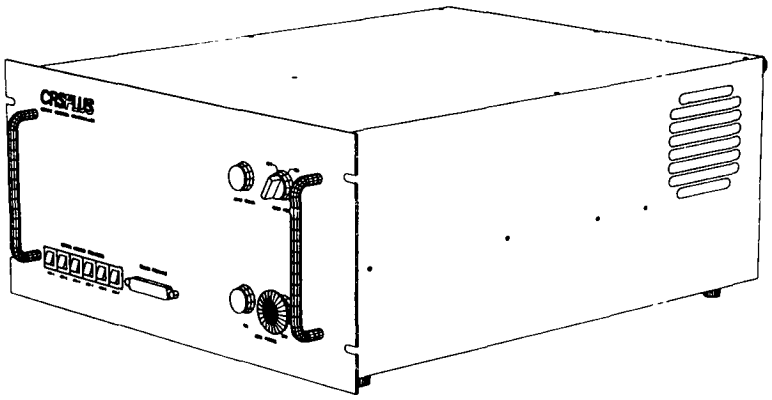
PIN#	FUNCTION
1	ENCODER GND
2	ENCODER 5 Vdc
3	CHANNEL A
4	NOT USED
5	CHANNEL B
6	NOT USED
7	NOT USED
8	CHANNEL Z
9	NOT USED
10	BRAKE ENERGIZE
11	MOTOR RETURN
12	MOTOR SUPPLY

FIGURE 1-3 Motor Connector

1-3 RSC-P8/PID8 ROBOT SYSTEM CONTROLLER

The A100/200 Series Robot System is controlled by one of the CRS Plus RSC Robot System Controllers. The difference between the two controllers is that the RSC-P8 uses proportional P-type axis cards while the RSC-PID8 uses PID-type axis cards. The former uses a simple logic axis card while the latter uses a microprocessor per axis card for additional control flexibility.

In this manual, whenever a difference is important to the technical discussion, the RSC-PID8 controller will be designated as A250 series and the RSC-P8 as the A150 series controller.



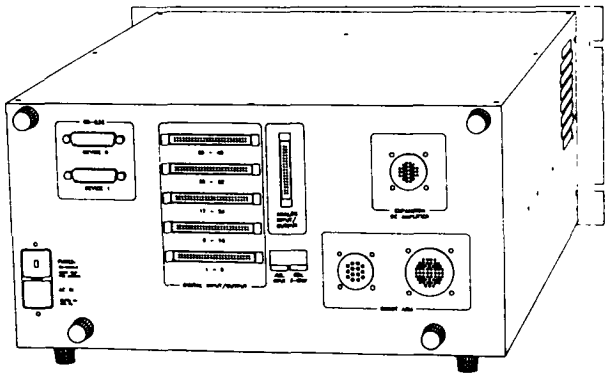


FIGURE 1-4 Robot System Controller - Front and Rear Views

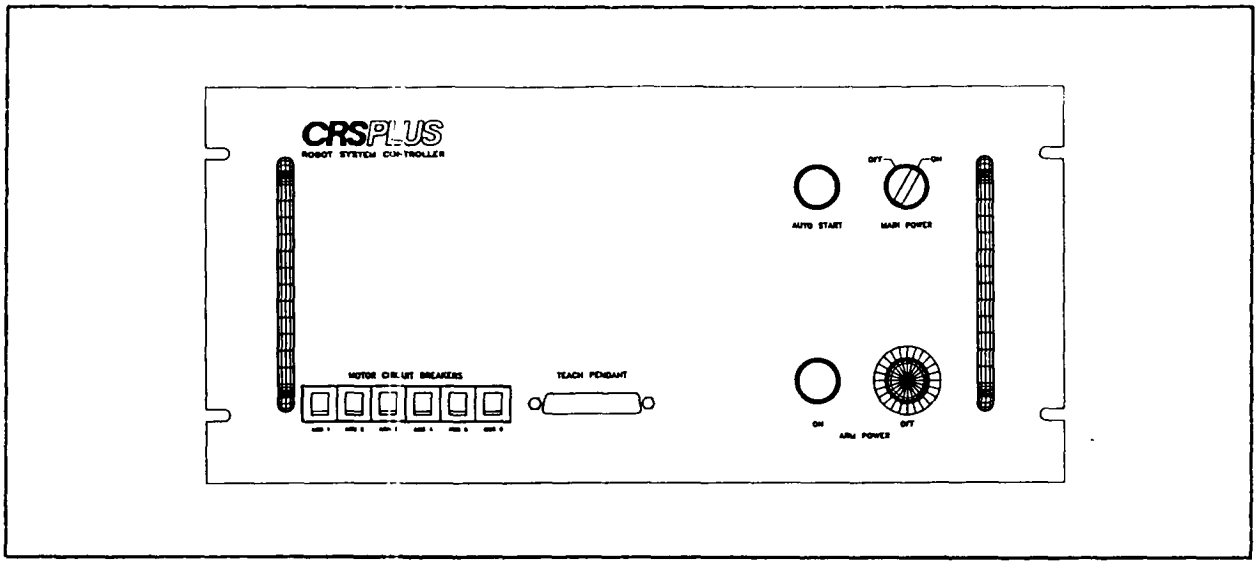


FIGURE 1-5 A100/200 Series System Controller (Front Panel)

Front Panel Features

1) **MAIN POWER Switch** - Turns power on to the robot computer to start operation. A light in the centre of the switch indicates power.

2) **AUTO START Switch** - Used in conjunction with the MAIN POWER Switch to permit automatic startup program execution.

3) **Motor Circuit Breakers** - Each amplifier output from the controller is protected by one of these breakers. The centre portion of the switch pops out showing a white underside when the circuit is interrupted. After a few seconds, the Breaker may be reset by pressing on the switch.

4) **Teach Pendant Connector** - DB-37 connector for hook up of the TEACH optional Teach Pendant (see section 1-4).

5) **ARM POWER ON Switch** - A lighted push-button which energizes power to the internal amplifiers, the servo gripper (if installed), the arm brake release circuits (A151/A251 Models only), and any external amplifiers (if used). The indicator light shows amber when energized.

6) **ARM POWER OFF Switch** - The red locking push-button removes power to the arm power circuits as described above. Can be used as an emergency stop in case of servo failure. The switch must be twisted to release before the arm can be energized again.

All front panel controls can be remotely located by using the remote front panel connection to the Front Panel Logic Board (see section 7-6).

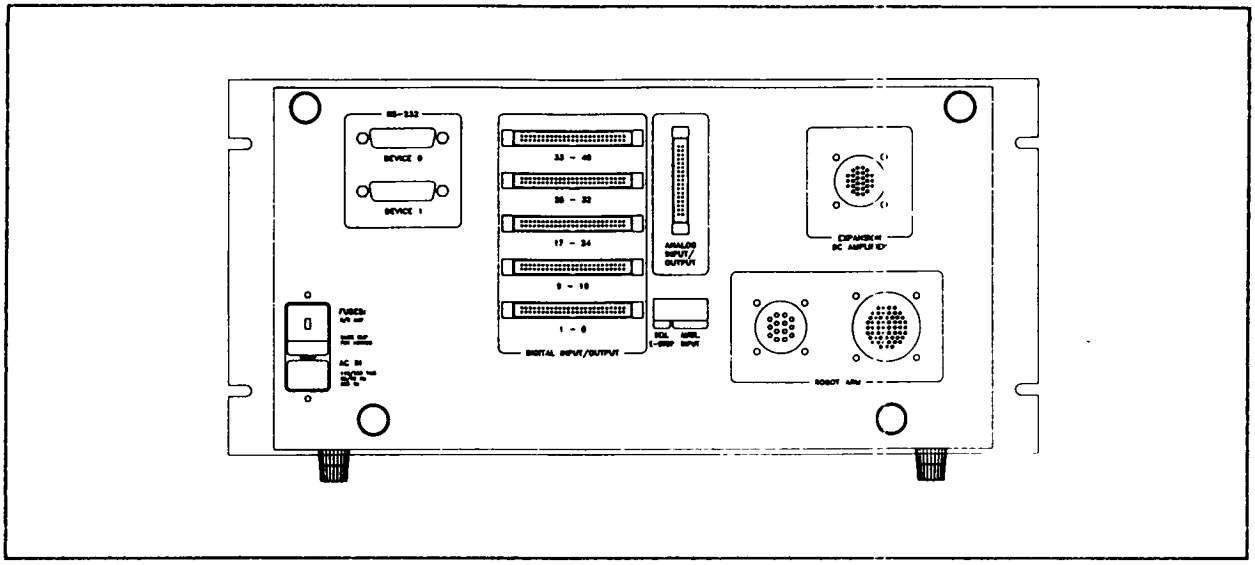


FIGURE 1-6 A100/200 Series System Controller (Back Panel)

Rear Panel Features

1) **AC IN / FUSES** - This contains the power cord receptacle, the voltage selector module (see section 3-4) and the AC line fuses.

2) **RS-232** - Device 0 and Device 1 are the two RS-232 serial ports which are used to connect the controller to external computer equipment such as terminals, computers and printers. See section 3-6 for inter-connection information.

3) **DIGITAL INPUT/OUTPUT Connectors** - These connectors interface the controller to user-programmable digital I/O. See section 3-8 for inter-connection information.

4) **ANALOG INPUT/OUTPUT Connector** - This connector interfaces the controller to user-programmable analog I/O. See section 3-9 for inter-connection information.

5) **E-STOP/AUX. INPUT Connection** - Pins one and two are used for connection to any external emergency stop (ARM-POWER OFF) switches. The remainder of the pins are used to support the special AUXILLIARY INPUT. The pin functions are listed in Table 1-1.

6) **EXPANSION DC AMPLIFIER Connector** - This connector carries command and feedback signals as well as AMP ENABLE signals from the mother board axis expansion slots (6 - 8) to an external amplifier. The pin-out for this connector is listed in section 7-11.

PIN	FUNCTION
1	REMOTE E-STOP
2	REMOTE E-STOP
3	+12 VDC SOURCE
4	+12 VDC SENSE
5	AUX INPUT
6	GND

TABLE 1-1 E-STOP/AUX. INPUT Connection Pin-out

7) **ROBOT ARM Connectors** - The cable between the arm and controller is functionally divided into two parts: the power connections and the signal connections.

The smaller of the two connectors on the rear panel carries the power connection. These include the outputs to the motors and the air gripper.

The larger connector carries all encoder and servo gripper signals as well as the brake signal.

1-4 TEACH PENDANT

The Robot System can be manually controlled by the teach pendant. The teach pendant provides control for all 8 axes of motion, at varying speeds set by a control on the pendant. It also permits teaching points, aborting programs, and and gripper control. The teach pendant also provides an LED for displaying error status.

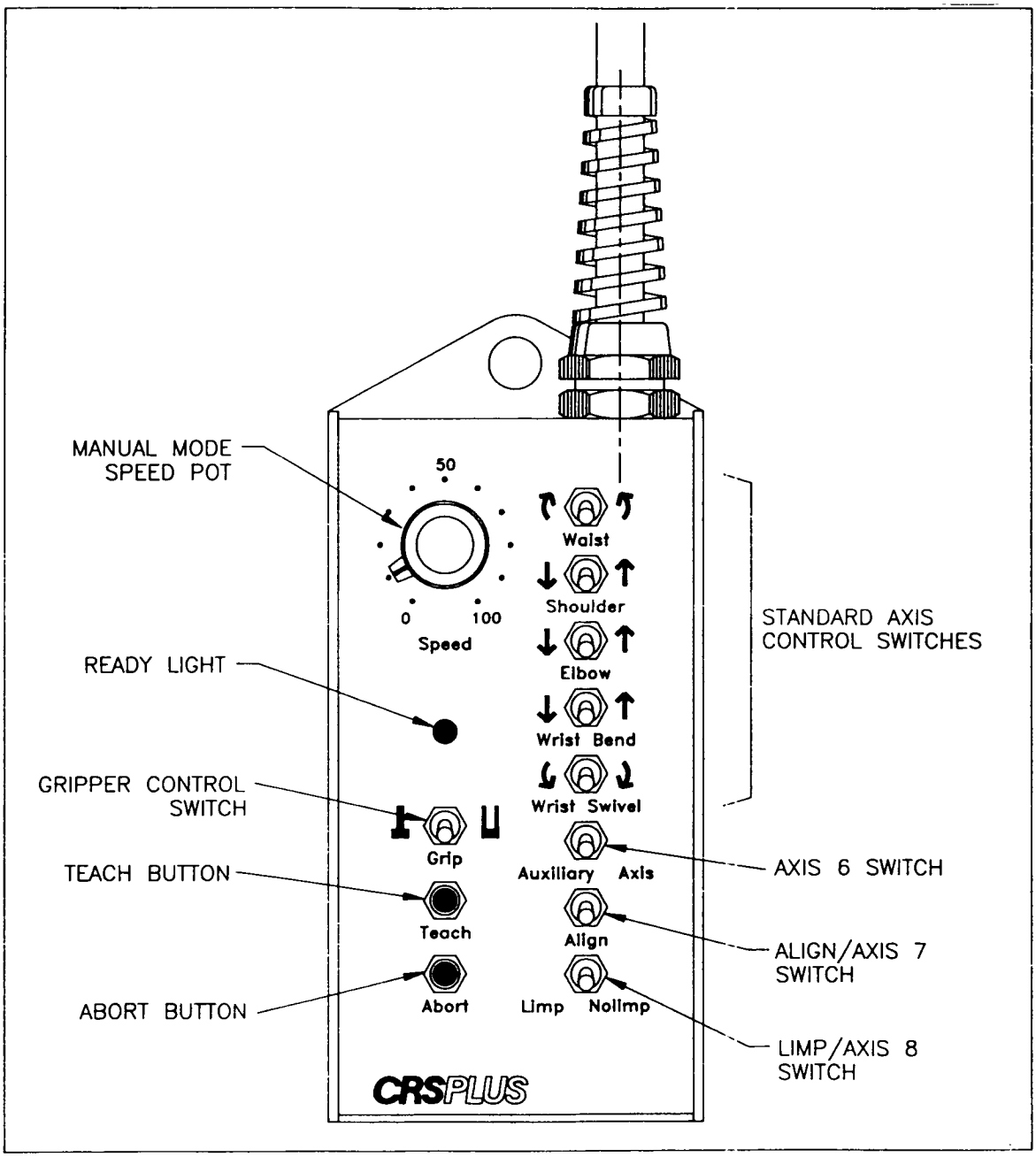


FIGURE 1-7 Teach Pendant

2-1 ROBOT ARM

Structure	Articulated 5 DOF
Drive motor	Permanent magnet DC Servo
Bearings	ABEC Class 1 - 0.375" ID
Max voltage	+/-25 Vdc
Max current	10.8 amps
Mech. time const.	11.62 msec
Max speed @ 25V	3600 rpm
Peak torque	100 oz-in
Brush life	8000 hours @ 1200 rpm
Transmission	
Waist rotate	Harmonic drive
Shoulder	Harmonic drive
Elbow	Harmonic Drive/chain
Wrist bend (pitch)	bevel-/spur-gear/chain
Tool roll	bevel-/spur-gear/chain/gear
Payload	kg
Maximum design	2.0
Full speed/acc	1.0
Reach - Waist to tool flange	22 inches
Reach (by link)	Inches
Base to shoulder	10
Shoulder to elbow	10
Elbow to wrist pivot	10
Wrist pivot to tool flange	2
Joint travel ranges	degrees
Waist rotate	+/-175
Shoulder	+110, -0
Elbow	+0, -130
Wrist bend (pitch)	+/-110

TABLE 2-1 Robot Arm Specifications (part 1)

Joint speeds at 100% program speed	rad/sec
A150 Series:	
Waist rotate	1.74
Shoulder3	1.08
Elbow	1.74
Wrist bend (pitch)	3.14
Tool roll	6.28
A250 Series:	
Waist rotate	3.05
Shoulder3	2.18
Elbow	3.05
Wrist bend (pitch)	3.14
Tool roll	6.28

Joint default acceleration rates	rad/sec^2
A150 Series:	
Waist rotate	5.45
Shoulder	5.45
Elbow	5.45
Wrist bend (pitch)	24.54
Tool roll	49.09
A250 Series:	
Waist rotate	12.93
Shoulder	12.93
Elbow	12.93
Wrist bend (pitch)	58.18
Tool roll	116.36

Position Feedback	Optical incremental encoders
Resolution	1000 pulse/rev.
Index	Marker pulse 1 per rev.
Output	Chnls A, B, Z sq.wave TTL

TABLE 2-1 Robot Arm Specifications (part 2)

Joint Resolution	deg
Waist rotate	0.005
Shoulder	0.005
Elbow	0.005
Wrist bend (pitch)	0.023
Tool roll	0.045

Joint Resolution	inches @ tool flange
Waist rotate	0.0019
Shoulder	0.0009
Elbow	0.0009
Wrist bend (pitch)	0.0008
Tool roll	0.0016

TABLE 2-1 Robot Arm Specifications (part 3)

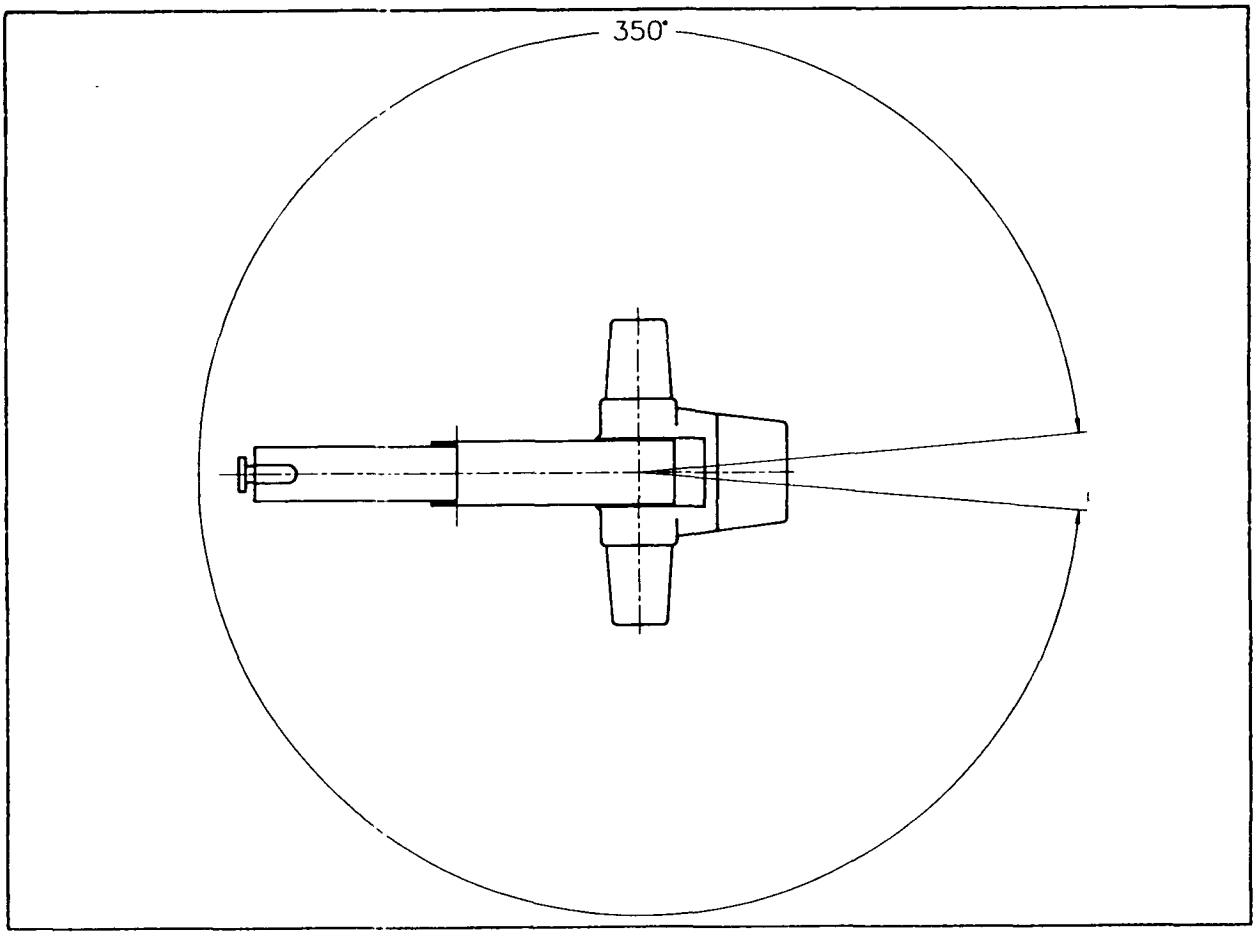


FIGURE 2-1 Workspace Plan

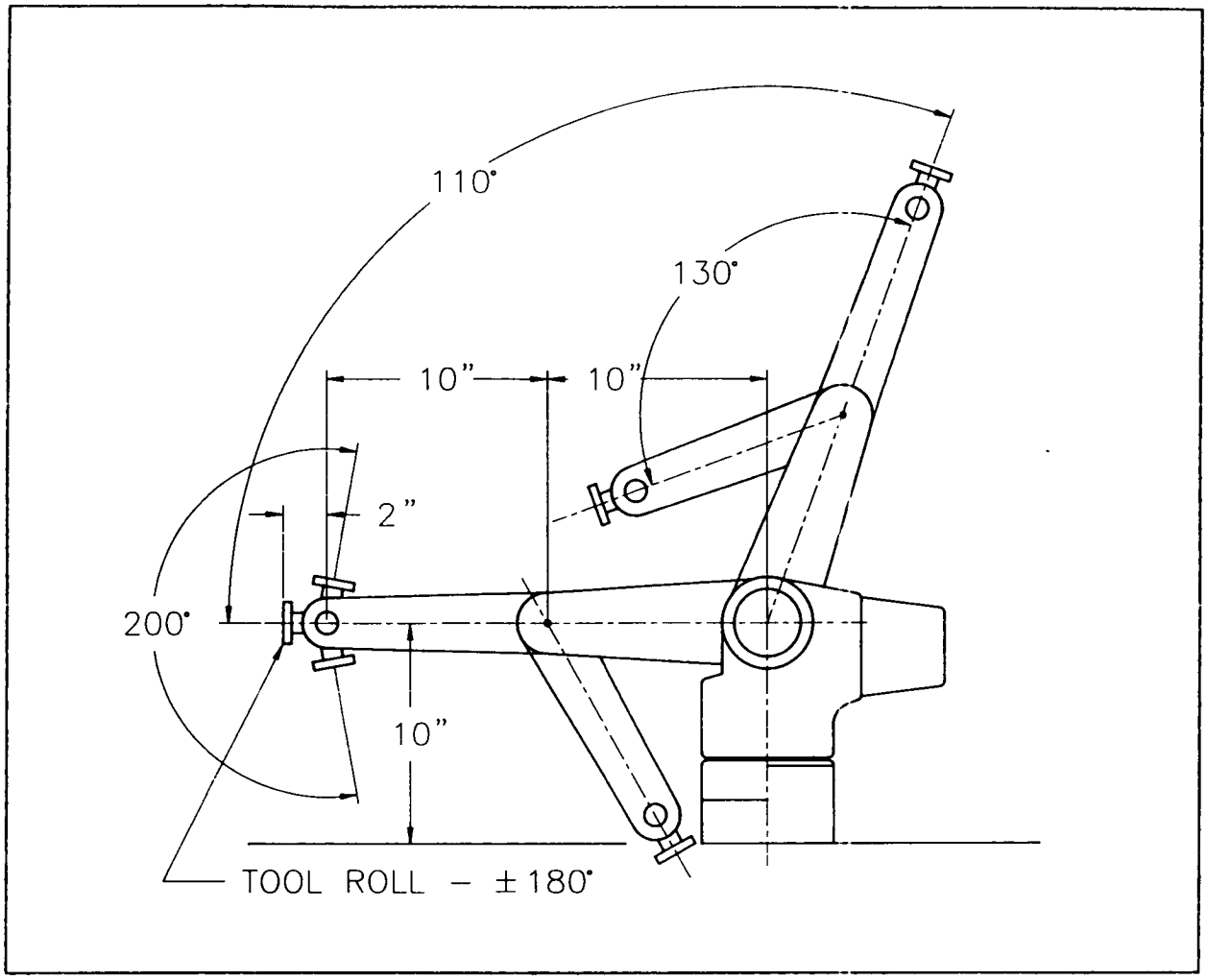


FIGURE 2-2 Workspace Elevation

Description	Specification
Control System:	
Microprocessor type	Intel 8086/NEC V30
Number of Axes	Eight (8) DC Servo Axes
Teaching System:	
Manual	Teach Pendant
Language	RAPL-II
Off-Line Optional	ACI Interface
Motion:	
Point to Point	Yes - Joint Interpolated
Straight Line	Yes - Possible on all moves
Continuous Path	Yes - 8 axis spline interp.
Position Detection	Digital Optical Encoders
Speed setting	0% to 150% Programmable
Interfaces:	
Communication	Dual RS-232C, ACI interface
User digital inputs	16 Standard TTL, 40 Optional
User digital outputs	16 Standard TTL, 40 Optional
Hardware Expansion	Peripheral Edge Connector
Programming:	
Language	RAPL-II
Editor	RAPL-II Line Editor
User Memory size	8 KByte Std, up to 64 K optional
Power Requirements	100-130 VAC, 50-60 Hz, 3A
Operating Ambient Temp.	0 to 50 degrees Celsius
Dimensions	19" wide x 21" deep x 7" high

TABLE 2-2 RSC-P8/PID8 Robot System Controller Specifications

2-3 TEACH PENDANT

8 Switches for Joint or Cylindrical Control LIMP/NOLIMP Control ALIGN switch 0 to 100 % Speed Control Abort Button Teach Button Ready Light Indicator

TABLE 2-3 Teach Pendant Specifications

3-1 UNPACKING AND INSPECTION

Upon receipt of your A100/200 Series Small Industrial Robot System check to ensure you have received all the components. The basic system is packaged for shipment in two foam-packed corrugated boxes. Any optional equipment shipped with the robot system will be found in at least one other box. See Table 3-1 for a description of the contents of the boxes. If any components are missing or damaged, contact your local distributor or CRS Plus immediately.

Description	Quantity
Box 1:	
Robot Arm	1
Robot Arm mounting bolts (installed in base)	4
Robot/Controller Cable	1
Power Cord	1
Box 2:	
Robot System Controller	1
Box 3 (When ordered):	
Teach Pendant	1
SRS-A100/200 Series Manual Set	1
** Other options **	

TABLE 3-1 A100/200 Series - System Components

Moving the Controller

The controller box is equipped with front panel handles to make it easier to lift. The rubber feet on the rear panel are designed to clear all connector receptacles once the plugs have been disconnected. The box can be tipped from its bottom feet to its rear feet without touching the box to the table or mounting surface. The handles aid in this manoeuvre. Once it has been tipped up, the controller can be lifted and carried using the front panel handles alone.

Moving the Arm

The arm is packaged in the box bolted to a small plywood pallet. The wrist is tied down to this pallet. The arm can be grasped under the lower link and lifted straight up to remove it from the box. Once out, carry the arm to its mounting area supporting its weight on the lower arm, steadied by holding on to the pallet. Remove the nuts from below the pallet to release it from the arm. Do not dispose of the bolts.

3-2 ROBOT BASE INSTALLATION

Install the robot base in accordance as follows:

1) Prepare a mounting platform. Drill holes in mounting platform using the detail shown in figure 3-1.

2) Firmly fasten the mounting platform to the floor to prevent vibration.

3) Mount the robot base on the mounting platform using the 3/8" diameter bolts (4) supplied with the arm.

4) For future precise relocation use 1/4" diameter dowel pins as shown in figure 3-1.

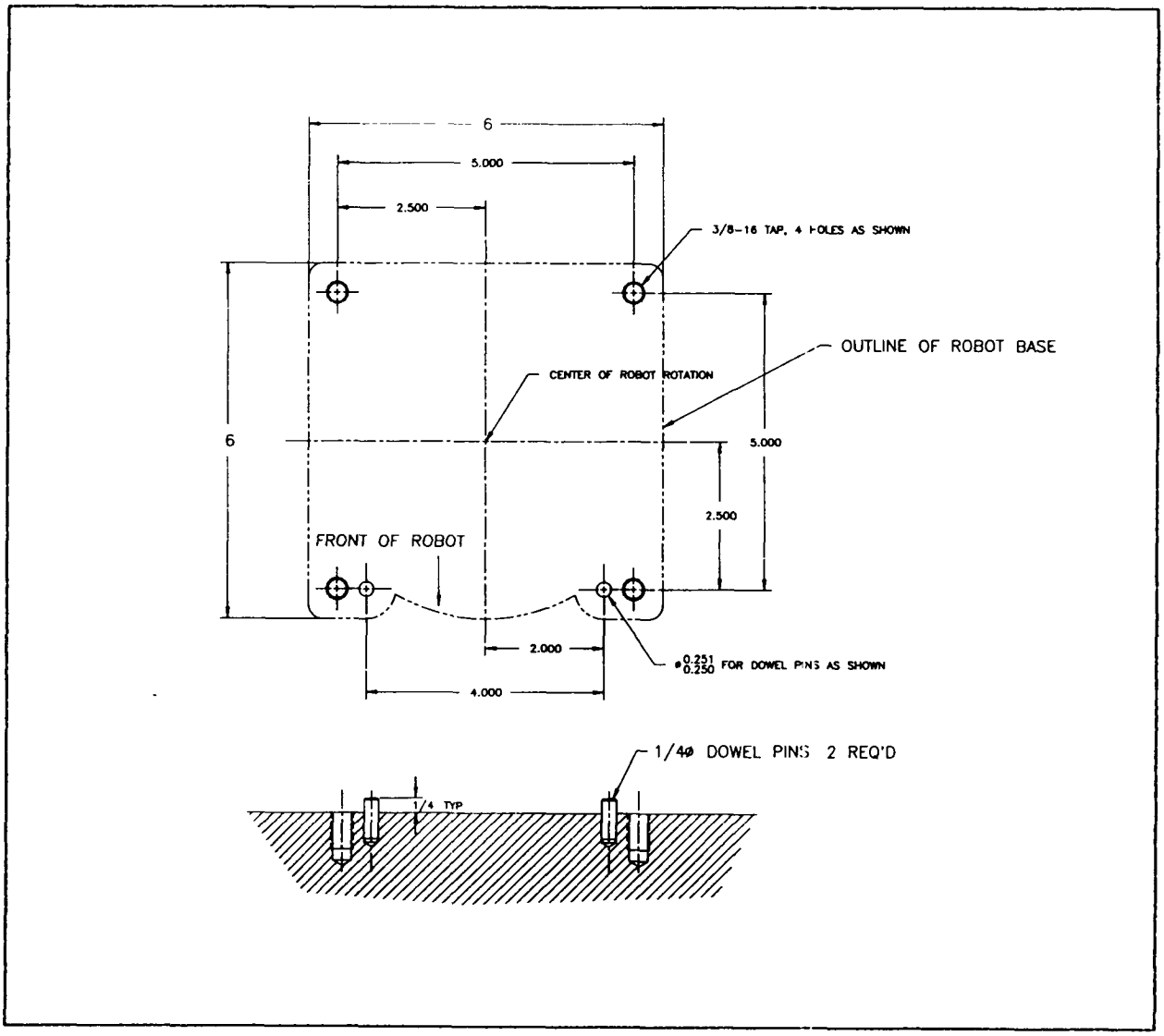


FIGURE 3-1 Robot Base Dimensions

3-3 GRIPPER INSTALLATION

Install the robot gripper as follows:

1) Fasten gripper to robot flange using 4, #10 socket head cap screws as Figure 3-2.

2) Connect air hose from pneumatic gripper into barbed fittings on the right side of the robot arm. The upper connector is pressurized after executing the CLOSE command (normally open) while the lower one is pressurized after the OPEN command (normally closed). For use with the SGRIP option, plug the electrical connector from the servo gripper into the receptacle on the left side of the robot arm.

3) For pneumatic operation, connect air supply to the 1/8-NPT air port at the rear of the robot base.

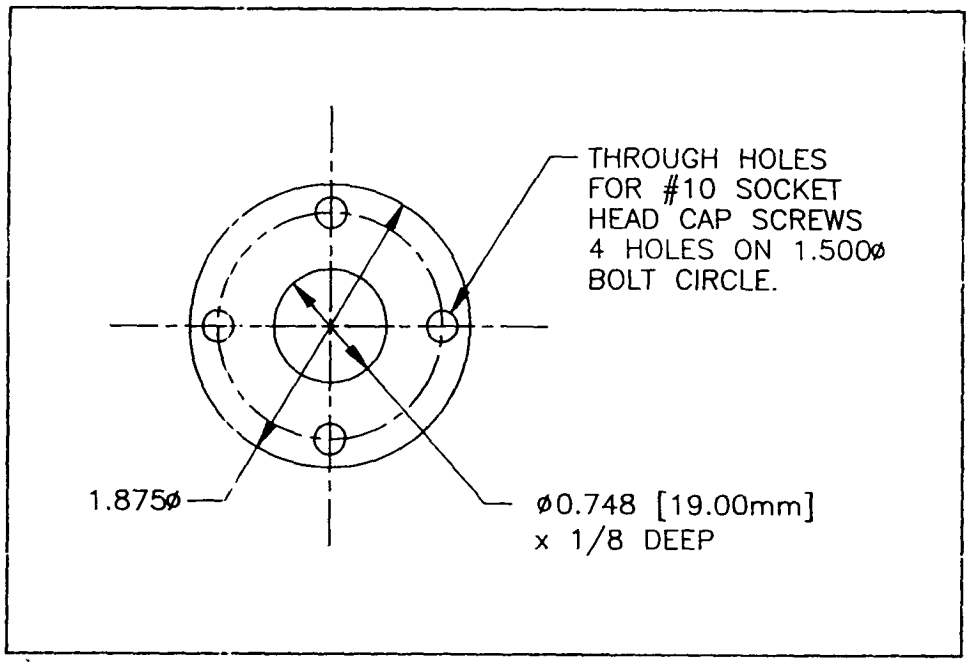


FIGURE 3-2 Robot Gripper Mounting Dimensions

APPENDIX B - ADVANCED COMMUNICATION INTERFACE

CONTENTS

TABLES

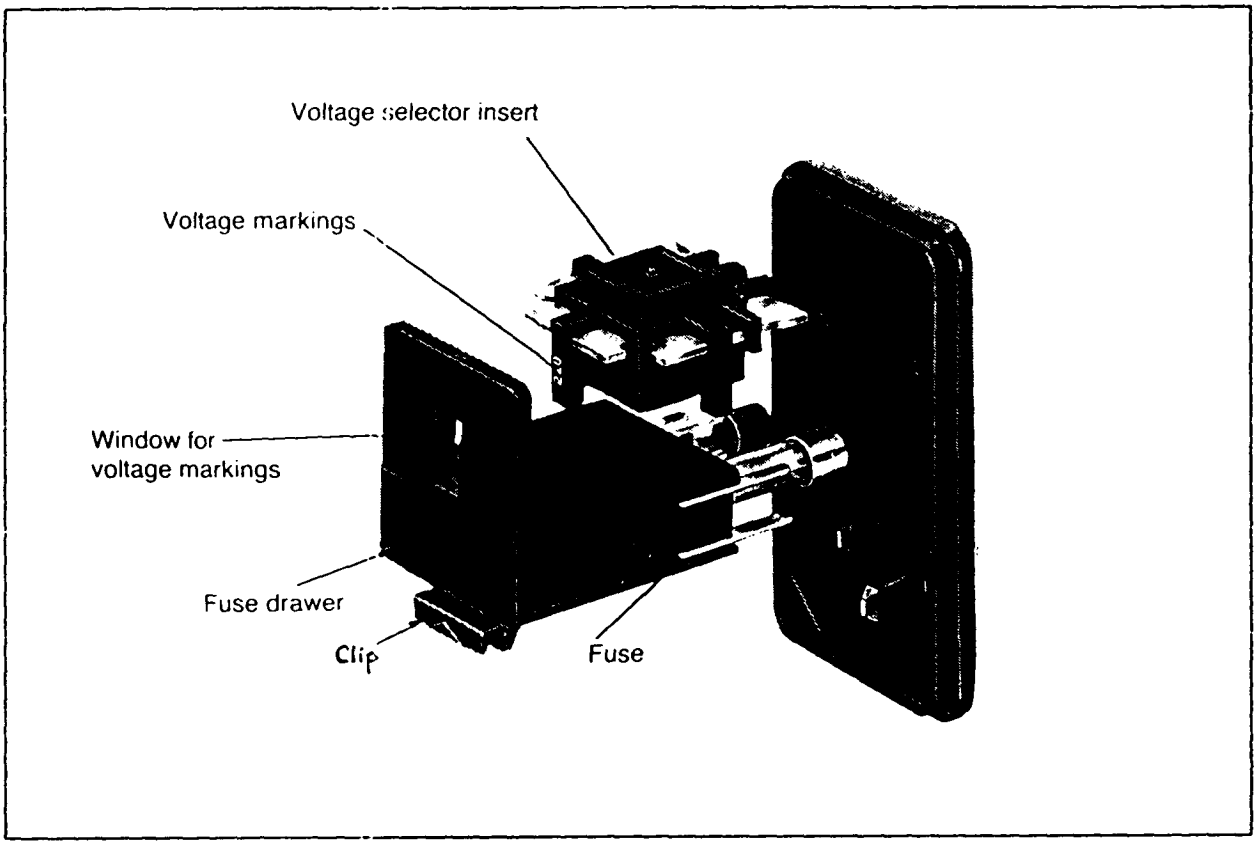


Figure 3-3 AC Voltage Entry Module - shown open.

3) 120 VAC or 240 VAC Main Power Setting. To change to 240 V, the setting of the AC module (see Figure 3-3) on the rear of the controller box must be reset and the AC fuses changed.

   1) Using a small screwdriver, press upward on the fuse drawer clip to release the fuse drawer.

   2) Slide the drawer out, making sure not to drop the fuses, and pull out the voltage selector insert.

   3) Insert the correct value of fuses into the rear of the fuse drawer. The correct values are: 5 AMP-250 Volt for 120 VAC operation
      3 AMP-250 Volt for 240 VAC operation

   4) Locate the correct voltage marking on one side of the selector. Re-insert the selector with the correct voltage value facing outward.

   5) Replace the fuse drawer pressing in until a distinct click is heard from the clip.

3-4 RSC-M1A ROBOT SYSTEM CONTROLLER INSTALLATION

Prior to installation of the controller, there are two set-up functions which should be checked: the operating AC voltage (120/240) and the setting for robot programming units (inches or mm).

1) The Country Kit:

Units are shipped from CRS in one standard configuration. That is set for operation at 120 VAC and using English (inch) programming units. All robots are also supplied with a "Country Kit" containing the correct standard power cord, the correct value of AC line fuses, and instructions as to how to set the system up to run in the correct programming units. Normally the system will be set up by a local distributor to conform with local standards.

2) Robot Programming Units

This feature is controlled by the setting of a DIP switch on the mother board. To access this switch, remove to lid of the controller and locate the Mother Board in the rear compartment. Figure 7-2 shows the layout of the Mother Board. The DIP switch is labeled in that figure as "DIP Switch 1".

To set the robot for INCH unit operation pole 1 of DIP Switch 2 must be set to OFF. Conversely, it must be set ON for metric. After changing the switch setting, a TEACH START must be performed to reset all software parameters to the new setting.

In addtion to the units function, there are other DIP switches on the mother board. The others will not normally be touched as they are set up when determining the model of controller and the type of EPROM used for the firmware. For the sake of completeness, the functions are listed here:

LABEL/POS'N	ON	OFF
DIP Switch 1 - SPDT Controller Model Select		
POLE 1	CLK1 - A150 Conf	CLK2 - A250 Conf
DIP Switch 2 (SW2) - 8 Pole Single-throw		
POLE 1	METRIC Units	INCH Units
POLES 2-8	Future Use	
DIP Switch 3 - DPDT EPROM Select		
POLE 1	C1 - 27256	C2 - 27512
POLE 2	C1 - 27256	C2 - 27512

3-5 CABLING

The basic cables used in the system are the AC power cable and the Robot-to-Controller cables (signal and power). These will be found packaged with the robot arm in a seperate envelope. During programming and initial installation it is recommended that a Teach Pendant (TEACH option) and terminal device be used.

1) Plug the AC power cord into three prong outlet. This will provide an earth ground for the equipment during the following hook-ups.

2) The main arm cable has the same connectors at each end. It is symmetrical and can be connected with either end at the controller. The connectors are keyed to prevent incorrect orientation. Connect the robot system cables to the rear panel of the controller. Minimize the bending stress on the panel connectors by strain relieving it to a solid object nearby. Connect the other end of the cables to the robot base.

3) Connect the teach pendant, if used, to the robot system controller front panel. The thumb screws in the connector may be used to provide a solid connection.

4) Connect your video terminal or PC to the robot system controller serial port #0 on back panel.

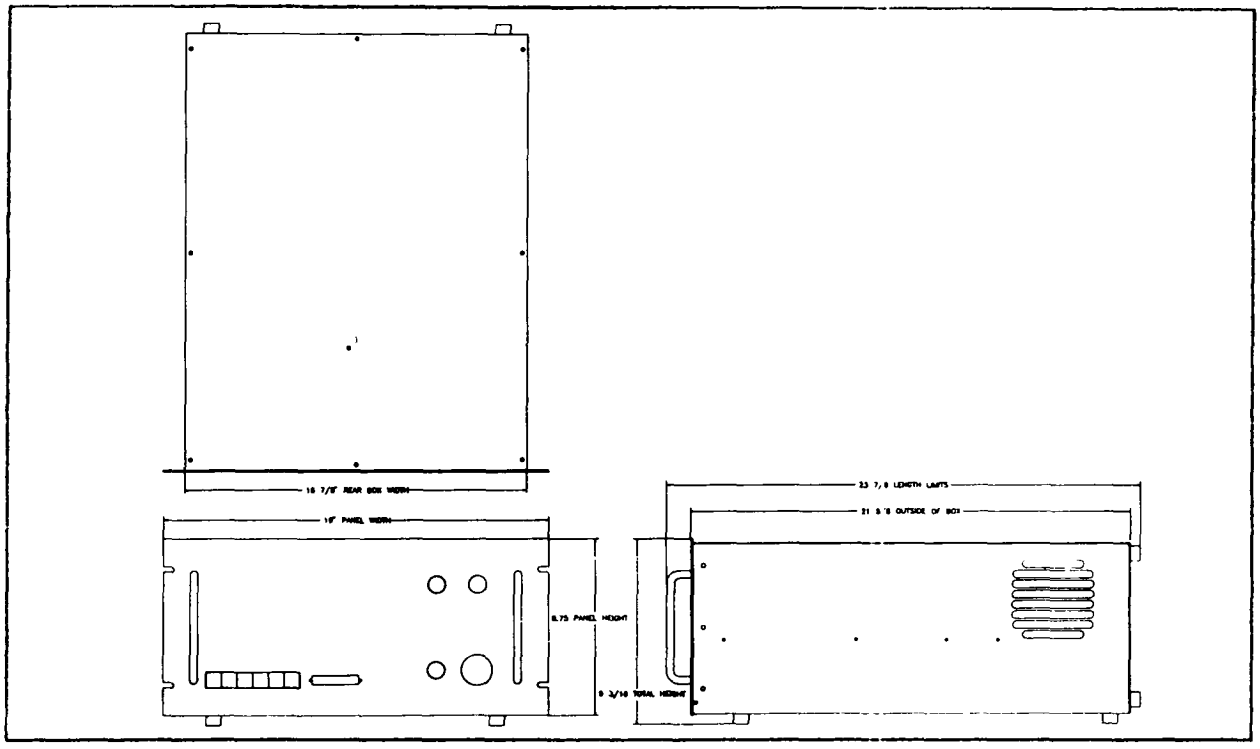


FIGURE 3-4 Controller Dimensions

A) Non-slide Mounting

The CRS controller is deisgned to be mounted either on a table-top or in a 19 inch standard rack. The box is equipped with rubber feet on both the bottom and the rear panel. It can be table-top mounted on its bottom feet if adequate cooling space is allowed around the system.

Install the robot system controller as follows:

1) Mount Controller into a 19" rack.

2) Allow no less than a 1" clearance for air flow in front of the grills on the right and left side of the enclosure.

3) It is recommended that the rear of the controller be supported where possible.

B) Slide Mounting

The CRS controller can be mounted with drawer slides to permit access for service and interconnection to other equipment. For instructions, refer to the drawer slide assembly manual.

3-6 SERIAL DEVICE CONNECTIONS (Continued)

B) Serial Printer Installation

Normally the printer would be connected to serial Device #1 on the rear panel. The pinout for the connection is shown in figure 3-6. Install the printer as follows:

1) Connect cable between printer and controller with the pinouts shown in figure 3-6.

2) Set printer RS-232 to 2400 baud, Full Duplex, No Parity, and XON/XOFF on.

3) Disable the ACI protocol for that port by issuing the RAPL-II PASSWORD and the @@RD command. After that all PRINT, PRINTI and PRINTV commands will send information to the printer.

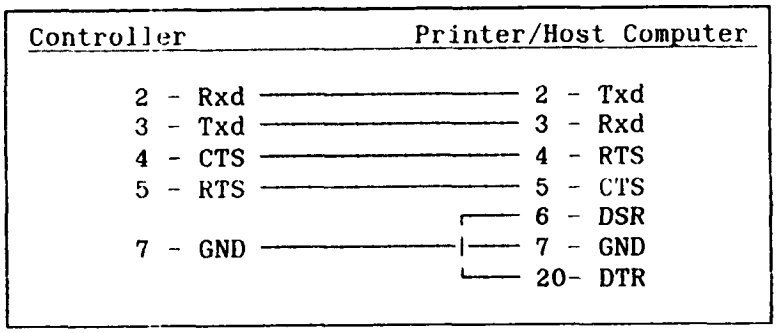


FIGURE 3-6 Printer Cable pin-out

C) Host Computer

A host computer may be connected as with the printer above. The same pin-out will be used in the cabling. If the ACI interface protocol is to be used, the @@RE command may be used to dedicate Device #1 to ACI use.

3-6 SERIAL DEVICE CONNECTIONS

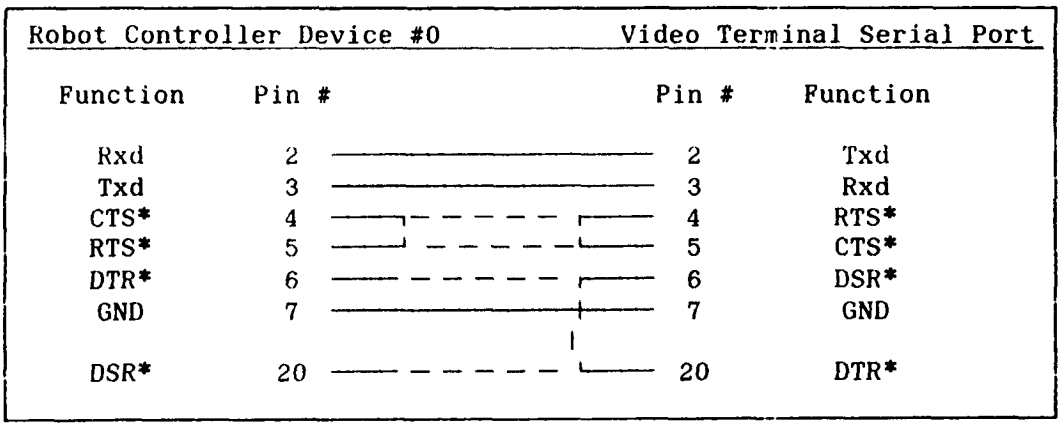
A) Video Display Terminal

Opreration of the robot for programming requires a terminal device to be attached. This may be a standard terminal operating therough a serial port or a personal computer running software which permits it to emulate a terminal. If running the ROBCOMM program from CRS, follow the interconnection instructions in the manual for that package. Install the video terminal as follows:

1) Connect cable to video terminal and controller with the pinouts shown in figure 3-5.

2) Set Video terminal to 9600 baud, Full Duplex, No Parity, and no auto line feed.

3) Install power cable to Video Terminal.



* May Be connected or "looped" as shown

FIGURE 3-5 Video Terminal Cable pin-out

Pin	Description	Pin	Description	Pin	Description
2-50	GROUND	2-50	GROUND	2-50	GROUND
1,49	Vcc	1,49	Vcc	1,49	Vcc
47	INPUT 1	47	INPUT 9	47	INPUT 17
45	INPUT 2	45	INPUT 10	45	INPUT 18
43	INPUT 3	43	INPUT 11	43	INPUT 19
41	INPUT 4	41	INPUT 12	41	INPUT 20
39	INPUT 5	39	INPUT 13	39	INPUT 21
37	INPUT 6	37	INPUT 14	37	INPUT 22
35	INPUT 7	35	INPUT 15	35	INPUT 23
33	INPUT 8	33	INPUT 16	33	INPUT 24
31	OUTPUT 1	31	OUTPUT 9	31	OUTPUT 17
29	OUTPUT 2	29	OUTPUT 10	29	OUTPUT 18
27	OUTPUT 3	27	OUTPUT 11	27	OUTPUT 19
25	OUTPUT 4	25	OUTPUT 12	25	OUTPUT 20
23	OUTPUT 5	23	OUTPUT 13	23	OUTPUT 21
21	OUTPUT 6	21	OUTPUT 14	21	OUTPUT 22
19	OUTPUT 7	19	OUTPUT 15	19	OUTPUT 23
17	OUTPUT 8	17	OUTPUT 16	17	OUTPUT 24
Connector 1-8		Connector 9-16		Connector 17-24	

Pin	Description	Pin	Description
2-50	GROUND	2-50	GROUND
1,49	Vcc	1,49	Vcc
47	INPUT 25	47	INPUT 33
45	INPUT 26	45	INPUT 34
43	INPUT 27	43	INPUT 35
41	INPUT 28	41	INPUT 36
39	INPUT 29	39	INPUT 37
37	INPUT 30	37	INPUT 38
35	INPUT 31	35	INPUT 39
33	INPUT 32	33	INPUT 44
31	OUTPUT 25	31	OUTPUT 33
29	OUTPUT 26	29	OUTPUT 34
27	OUTPUT 27	27	OUTPUT 35
25	OUTPUT 28	25	OUTPUT 36
23	OUTPUT 29	23	OUTPUT 37
21	OUTPUT 30	21	OUTPUT 38
19	OUTPUT 31	19	OUTPUT 39
17	OUTPUT 32	17	OUTPUT 40
Connector 25-32		Connector 33-40	

FIGURE 3-8 Digital I/O Pinout Description

3-7 DIGITAL I/O INSTALLATION (OPTIONAL)

Install the DB/16R or equivalent as follows:

1) Connect the ribbon cable to the side of the rack. the connector is keyed to ensure correct alignment.

2) Mount the DB/16R in the 19" rack above or below the controller.

3) Connect cable to one of the controller rear panel connectors labeled "DIGITAL INPUT/OUTPUT". Generally, the lowest is used first. Only the two lower ones (1-8 and 9-16) are supported in standard configuration. If more I/O is needed above these, the COMBO card (COMBO/32) must be used.

4) If not using the DB/16R contact your distributor for information on compatibility with other isolators or systems. Use Ansley part number 609-5030 to connect to the controller. The pinouts needded are shown in Figure 3-8 and 3-9.

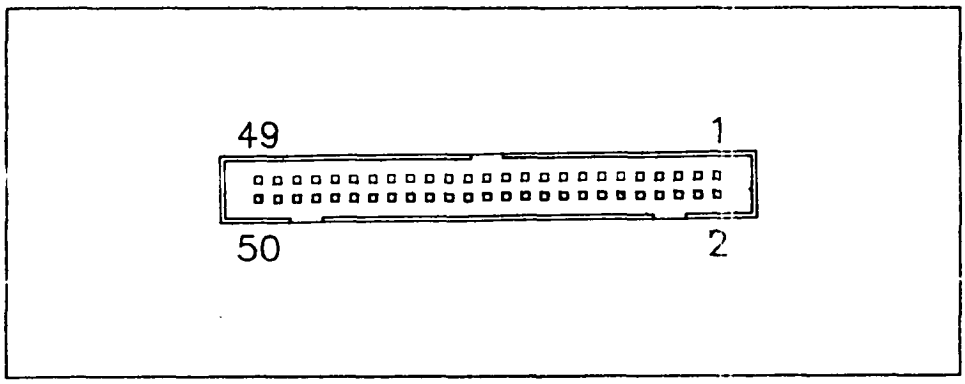


FIGURE 3-7 Pin orientation for each digital I/O connector

5-1 POWER UP SEQUENCE

A) Startup for Programming and Set-up

1) Turn on Video Terminal or PC. If using ROBCOMM, enter the TERMINAL EMULATE mode.

2) Turn on Robot system controller. Observe the Sign-on message. If it does not appear, follow the trouble-shooting flow-chart in APPENDIX H.

3) Manually move robot into the upright position and hold in position. For models equipped with brakes (A151 or A251), the arm power must be on to release the brakes. Therefore, to move the arm into position, the system must be in MANUAL or LIMP mode. Refer to the RAPL-II Reference manual or the Tutorial manual for further details.

4) Turn on arm power.

5) You are now ready for the Homing sequence (SECTION 5-2).

B) Auto-Startup (after programming is complete).

1) Ensure the arm is in its correct starting position. Generally it will be nested in a homing bracket. If it is not, place it correctly before proceeding.

2) Press and hold the Auto-start push-button.

3) Turn on the main power, continuing to hold the AUTO-START push-button until the green light in the push-button comes on. This is an indication that the AUTO_ST program is running. If it fails to come on and the light on the teach pendant goes out, no AUTO_ST program was found by the controller.

4) Turn on the ARM POWER. The arm should now proceed to its homing sequence as programmed in the AUTO_ST routine.

3-8 ANALOG INPUT INSTALLATION (OPTIONAL)

Analog I/O is available only with the optional COMBO/32 expansion card. Connect analog inputs to rear panel of the controller using Ansley 609-3430 or equivalent. Pinout for analog connection is shown in figure 3-10. Figure 3-11 shows the pinout for the remainder of the analog I/O connector if this card is installed.

The analog inputs are for 0 to 5 Vdc and use 8-bit resolution. Analog outputs are +/- 10 Vdc and use 8 bit resolution.

Pin	Description	Pin	Description
11	GROUND	22	ANALOG IN 15
12	GROUND	23	ANALOG IN 16
		24	ANALOG IN 17
13	ANALOG OUT 1	25	ANALOG IN 18
14	ANALOG OUT 2	26	ANALOG IN 19
		27	ANALOG IN 20
15	GROUND	28	ANALOG IN 21
16	GROUND	29	ANALOG IN 22
		30	ANALOG IN 23
17	ANALOG IN 10	31	ANALOG IN 24
18	ANALOG IN 11	32	ANALOG IN 25
19	ANALOG IN 12		
20	ANALOG IN 13	33	GROUND
21	ANALOG IN 14	34	GROUND

FIGURE 3-9 Analog Input pin description with optional COMBO expansion card.

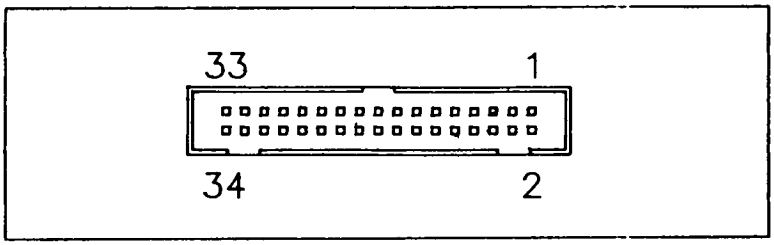


FIGURE 3-10 Analog input connector pin location diagram

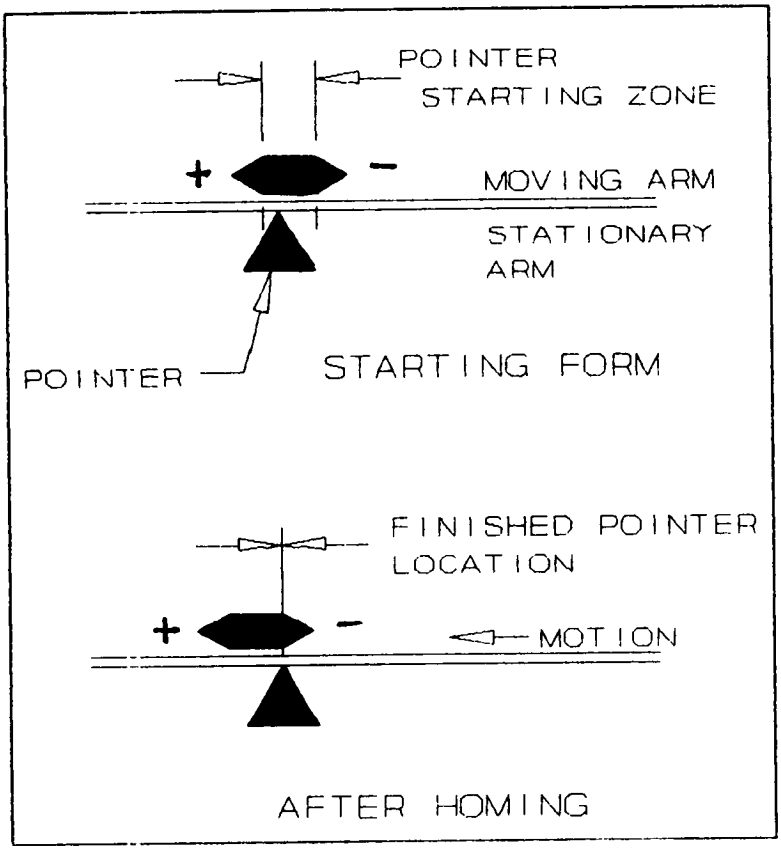


FIGURE 5-1 Homing Marks

6-1 ROBOT ARM CONSTRUCTION

The A100/200 Series robot system has a five axis (joint) arm. The five joints as shown in figure 6-1 are the waist (1), shoulder (2), elbow (3), wrist bend (4), and wrist roll (5).

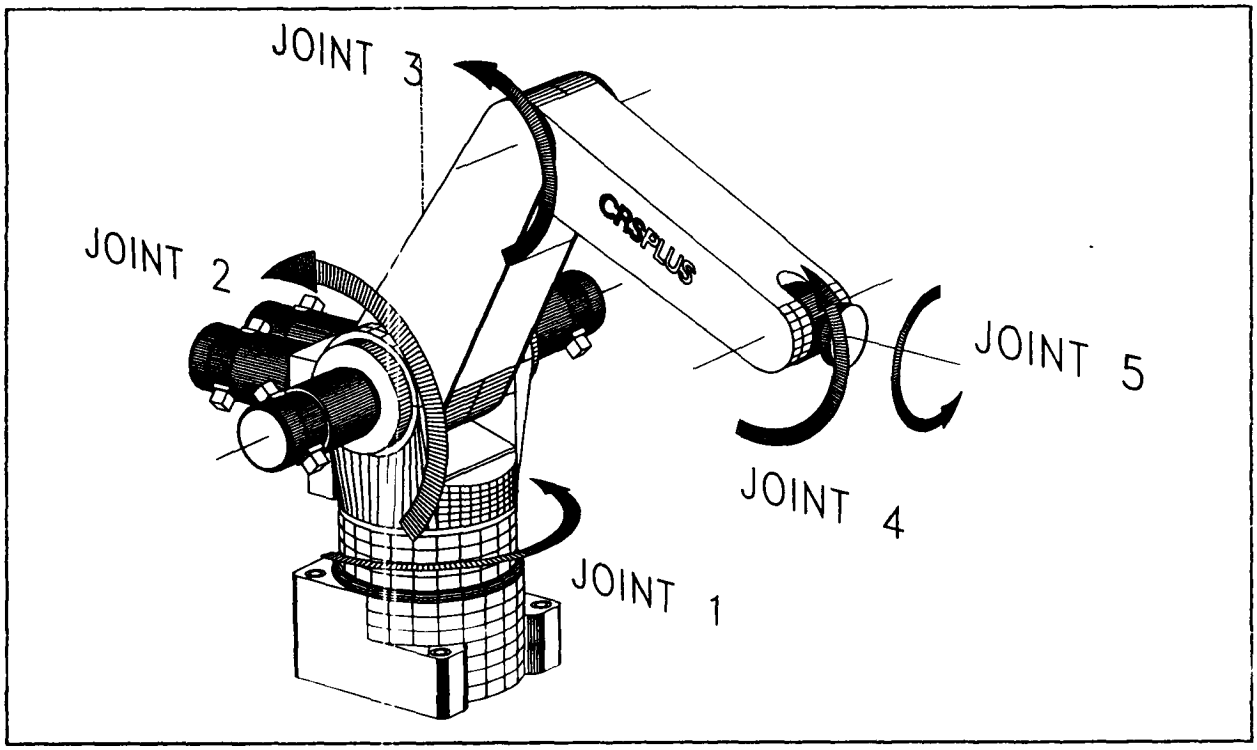


FIGURE 6-1 Robot Joints

The workspace of the robot is defined by the type of joints, the length of the links, and the range of motion of each joint. This data is contained in Table 6-1 and shown graphically in Figure 6-2.

The arm is constructed of high tensile aluminum alloy components. It features side panels held rigid by crossmembers and linked by a stressed aluminum skin. This construction technique gives light weight while contributing to the rigidity of the arm which in part allows for the high speed and accuracy of the system.

Each joint is supported by precision-grade sealed ball bearings. These are all conservatively rated as to load and speed conditions, and will not need servicing for the life of the machine.

The base of the arm and the supports for the moving arm components are cast aluminum which creates a solid foundation for the arm without adding significant weight to the structure.

5-2 HOMING SEQUENCE

1) Do the start-up sequence listed above (section 5-1 A).

2) Type >>**MAN**UAL<cr> at keyboard to enter manual mode.

*Future 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 the user must type the "STA" and RAPL-II completes it with the "US" before the user types the <cr>. In this way, users who prefer to turn off the HELP mode can see the complete command and others know how many letters to type.

3) Starting with Joint #1, move all five joints to the homing mark positions. Position each joint such that the homing marks are as shown in Figure 5-1 (the arrow point is within the homing range).

4) Enter >>**HOME** at keyboard. The controller will ask you to confirm that you have moved the robot arm into the starting home range.

5) Re-check the marks as above and enter **Y<cr>**.

6) The arm will move joint by joint into its reference position and display on your screen the status as each joint homes successfully. If a error occurs in any of the joints return to step 1) and repeat the complete procedure. If a error occurs in three (3) consecutive attempts contact your local distributor or CRS Plus.

7) Visually inspect each homing marker. After successful completion of the HOME sequence all markers should point directly at each other as shown in Figure 5-3. If the arm did not home correctly return to step 1) and repeat the complete procedure.

8) If the arm did HOME correctly and you are now ready to start developing and executing programs.

For more detail on the programming and start up of the robot system refer to the Tutorial and RAPL-II Programming manuals.

6-2 DRIVE MOTORS (Continued)

A schematic of the operation of the optical encoder is seen in figure 6-3. The optical encoder is as its name inplies, an optical device using a pair of idsks etched, with a fine grating. One disk is fixed, the stator, and the other, the rotor, is fixed to the motor shaft so that it rotates. As the rotor rotates past the stator, light can alternatively pass and not pass through the gratings on the pair of disks. A photocell picks up this light signal and converts it to a voltage. This signal is wave-shaped into a train of pulses which are directly related to the position of the shaft. The encoders use two stators 90 degrees of phase apart are used to determine the direction of rotation.

The encoders used in the A100/200 Series are incremental which means that they only count up or down from the current location. The A100/200 Series encoders have 1000 pulses per revolution.

6-3 DRIVES AND TRANSMISSIONS

The A100/200 Series robot system uses Harmonic drives to reduce motor speed to link speed with a corresponding increase in torque.

The Harmonic Drive can be seen in figure 6-4. The output torque rating is 110 in.lb. continuous.

The units used in the A100/200 Series have a 72:1 reduction ratio. Joints 1 and 2 have the flexspline coupled directly to the moving member. In joint 3, the output is transmitted through a preloaded 1/4" pitch roller chain to the link. These drives give accurate response with virtually no backlash.

The Harmonic drive has a further advantage in that it is back-drivable. This allows the robot to be manually positioned when the servos are limped or powered off.

The wrist drive is a more complex transmission. The wrist motors drive a series of preloaded 1/4" pitch chains which transmit the motion to the wrist. The reduction in the drive is through a pair of adjustable gear sets, first bevel and then spur. Joint four is directly coupled to the chain sprocket. Joint 5 (the tool flange) is driven by a pair of bevel gears, driven by the sprocket of the chain drive.

For adjustment of the transmissions, refer to Appendix C.

7-1 INTRODUCTION

This section describes the RSC-M1A controller hardware. Table 7-1 describes the various modules.

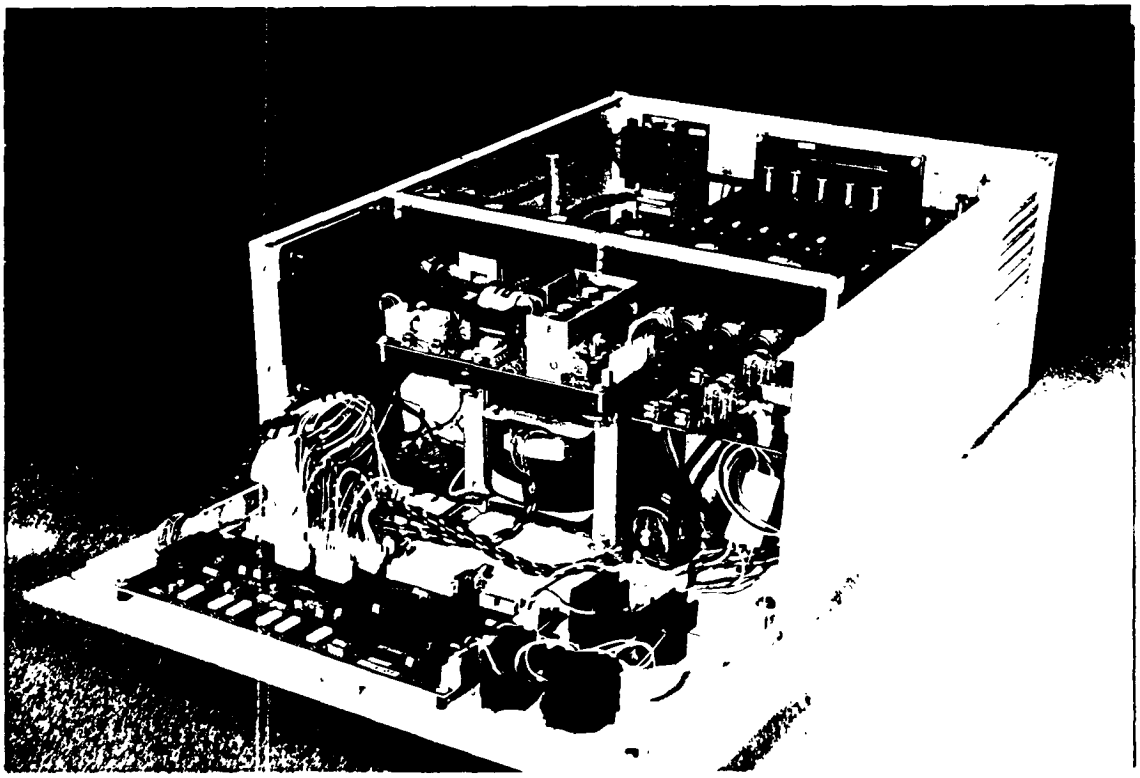


FIGURE 7-1 A100/200 Controller (Lid removed and front panel dropped)

The controller is designed to be servicable. The controller box is designed in two compartments: the front contains all the power supply and AC circuits, while all control electonics are in the rear compartment. A bulkhead which supports the arm amplifiers forms the break between the two.

Notice in Figure 7-1 that the front panel drops to allow access to the front compartment of the controller box. This is accomplished by removing the three #10 button head screws on each side of the front of the main box. The smaller #8 button-head screws at the bottom of each side forms the "hinge" for the panel. Do not remove these screws. Once the front panel has been dropped, the front pan which holds the isolation transformer, AC conditioning circuitry, computer power supply, and the arm power switching/filter board can be removed. To do this, undo the main AC connector and all harnesses between the pan and the front panel, and remove the nuts on the studs at the corners of the pan.

6-1 ROBOT ARM CONSTRUCTION (Continued)

BASE ROTATION	+/- 175 Deg.
SHOULDER ROTATION	-110, -0 Deg.
ELBOW ROTATION	-0, -130 Deg.
WRIST BEND	+/- 110 Deg.
WRIST ROLL	+/- 180 Deg.

TABLE 6-1 Robot Joints: Ranges of Motion

6-2 DRIVE MOTORS

The Motors used in the A100/200 Series system are permanent magnet DC servo Motors. These are constructed to industrial standards and fitted with incremental optical shaft position encoders.

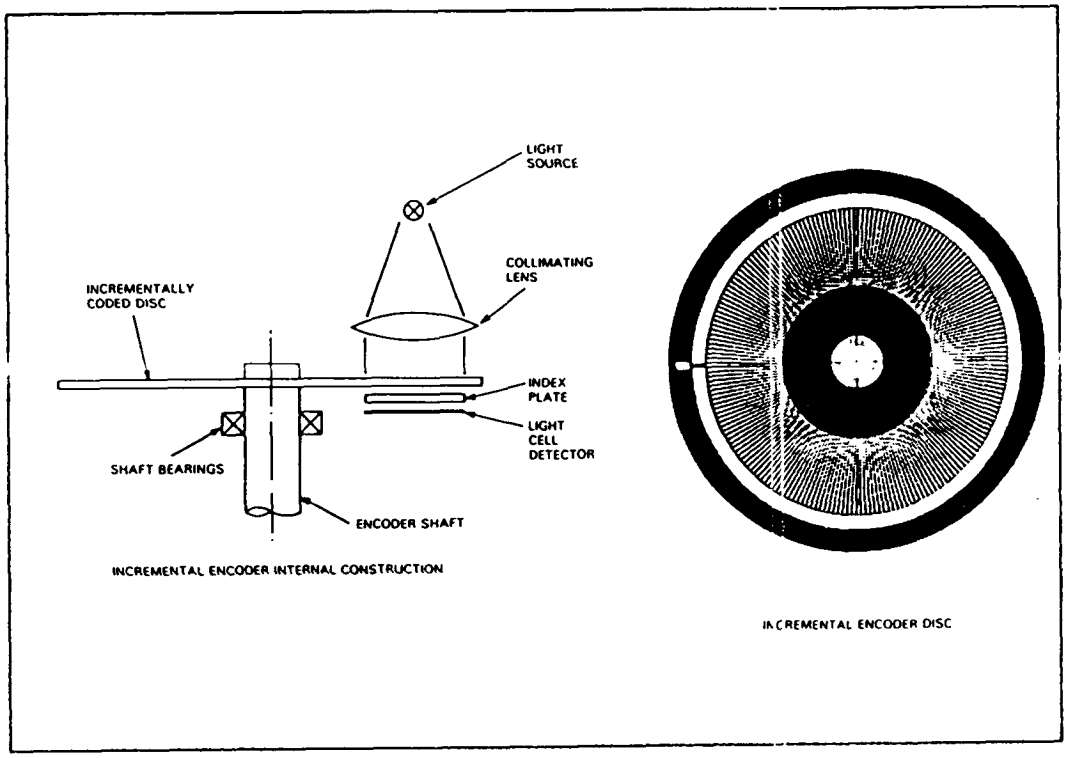


FIGURE 6-3 Optical Encoder Schematic

7-2 MOTHERBOARD

The Motherboard as shown in figure 7-2, is the largest printed circuit board (P.C.B.) in the CRS RSC-M1A controller. Table 7-2 describes the Motherboard connectors and major features.

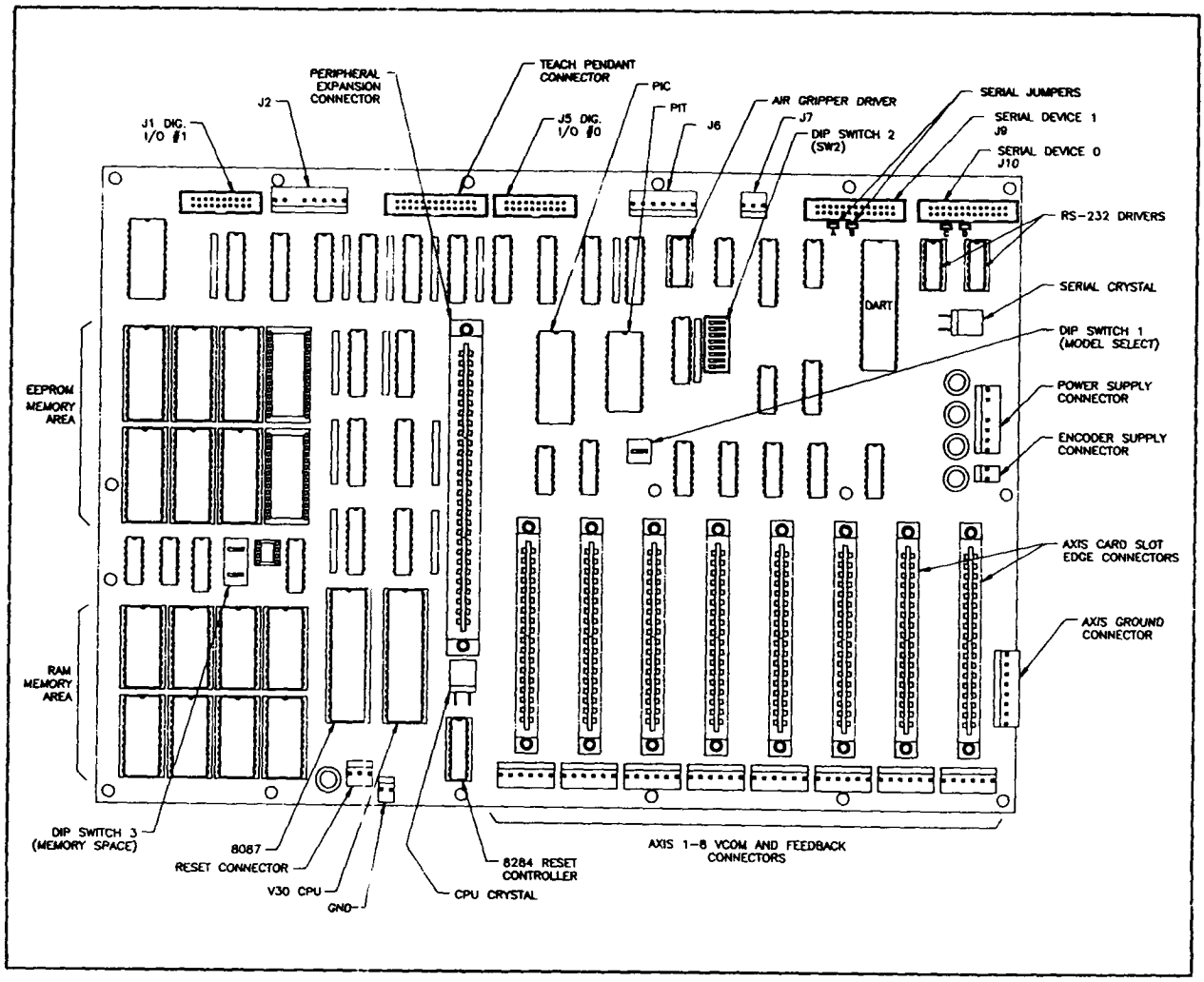


FIGURE 7-2 Motherboard P.C.B.

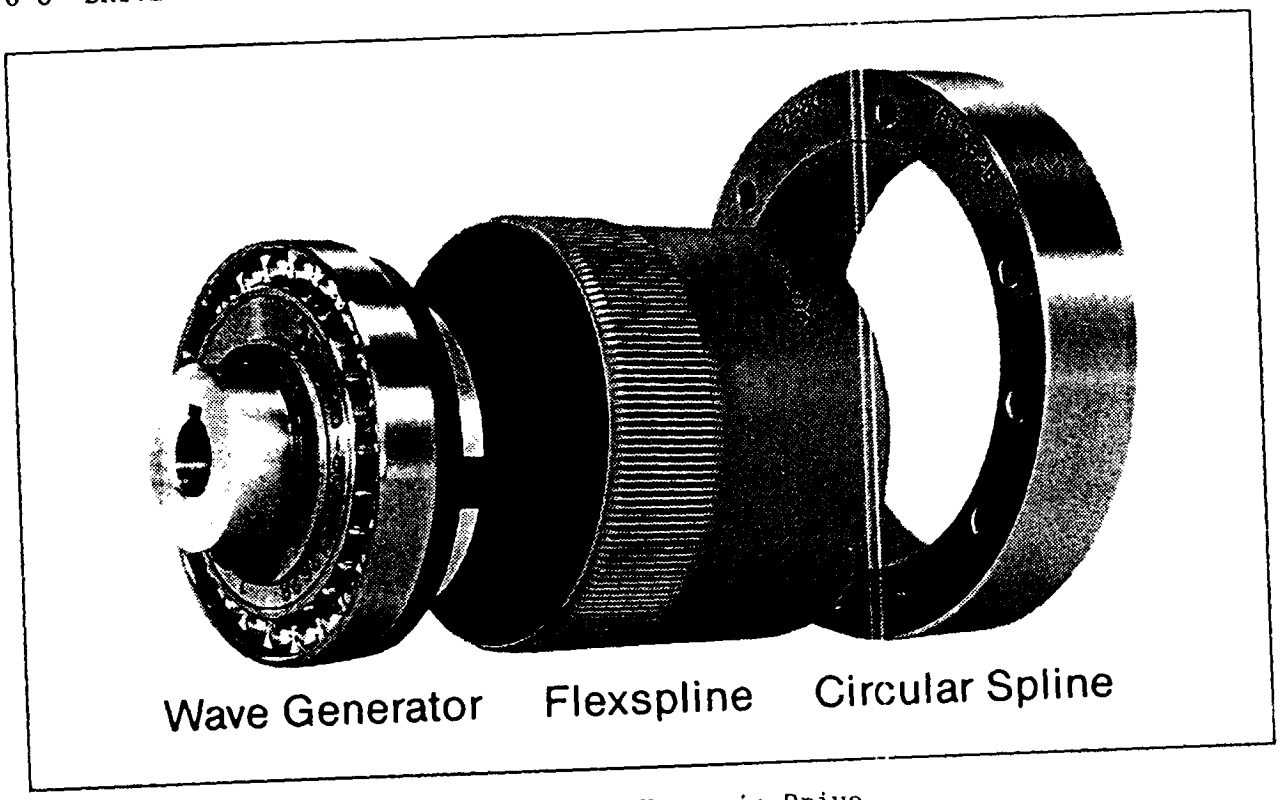
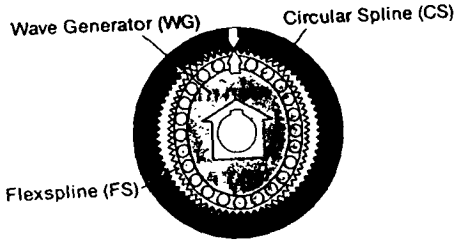
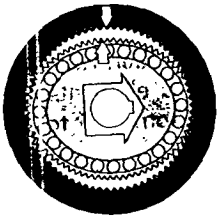


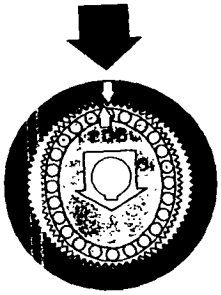
FIGURE 6-4 Harmonic Drive



1 The Flexspline (FS) is slightly smaller in diameter than the Circular Spline (CS), resulting in its usually having two fewer teeth on its outer circumference. It is held in an elliptical shape by the Wave Generator (WG) and its teeth engage with the Circular Spline across the major axis of the ellipse.



2. As soon as the Wave Generator starts to rotate clockwise, the zone of tooth engagement travels with the major elliptical axis.



3 When the Wave Generator has turned 180° clockwise, the Flexspline has regressed by one tooth relative to the Circular Spline.

4 Each turn of the Wave Generator moves the Flexspline two teeth backwards on the Circular Spline.

Common to Both A150 and A250 series:
- Level 3 is used for scanning the digital I/O every 40 mSec.
- Level 4 is to support the TEACH button on the teach pendant.
- Level 5 is to support the ABORT button on the teach pendant.
- Level 6 is to support the serial channel 0.
- Level 7 is to support the serial channel 1.

3) A programable interval timer (PIT) is used to generate fixed time intervals. The PIT has 3 timers which are used as follow:

A150 Series:
- Timer 0 generates the time base for the position loop of (~3.8 mSec).
- Timer 1 generates the time base for the command generation (variable)
- Timer 2 generates the time base for the I/O scanning (~40 mSec).

A250 Series:
- Timer 0 generates the time base for the watch dogs (~3.8 mSec).
- Timer 1 generates the time base for the command generation (variable).
- Timer 2 generates the time base for the I/O scanning (~40 mSec).

4) A Dual Asynchronous Receiver/Transmitter (DART) which controls two serial channels with the RS-232 format having a programmable baud rate of between 50 and 19,200 bps.

5) DIP Switches:

Three DIP switches are used to set the configuration of the mother board for various operational modes:

LABEL/POS'N	ON	OFF
DIP Switch 1 - SPDT Controller Model Select		
POLE 1	CLK1 - A150 Conf	CLK2 - A250 Conf
DIP Switch 2 (SW2) - 8 Pole Single-throw		
POLE 1	METRIC Units	INCH Units
POLES 2-8	Future Use	
DIP Switch 3 - DPDT EPROM Select		
POLE 1	C1 - 27256	C2 - 27512
POLE 2	C1 - 27256	C2 - 27512

The rear pan can also be removed by undoing harnesses and removing the four nuts which secure it.

Item	Quantity	Module Number	Location	Description
1	1	SEC-13-700	Card Cage	Mother Board
2	5	SEC-13-701	Card Cage	P-type Axis card
			or	
2	5	SEC-13-702	Card Cage	PID-type Axis card
3	2	SEC-13-703	Rear Pan	D.C. Amplifier Module
4	1	SEC-13-704	Front Panel	Front Panel Logic Board
5	1	SEC-13-705	Rear Panel	Digital I/O Connection Board
6	1	SEC-13-708	Front Pan	Computer Power Supply
7	1	SEC-13-716	Front Pan	Arm Power Supply
8	1	SEC-13-709	Front Pan	Arm Power Filter Board
9	1	SEC-13-7XX	Rear Panel	Encoder Connector Board
10	1	SEC-13-7XX	Rear Panel	Amp Expansion Board

TABLE 7-1 A100/200 Series Controller Desciption

7-3 DC AMPLIFIER MODULE

The D.C. Amplifier Module as shown in Figure 7-3 contains three (3) separate amplifiers, each supplied with +/- 26 VDC. The signal to each amplifier is +/- 10 VDC. The output is the motor voltage of +/- 20 VDC. at 2 amperes each.

Each module has an adjustable gain in the range of x1 to x2. It has no phase reversal and a low crossover distortion.

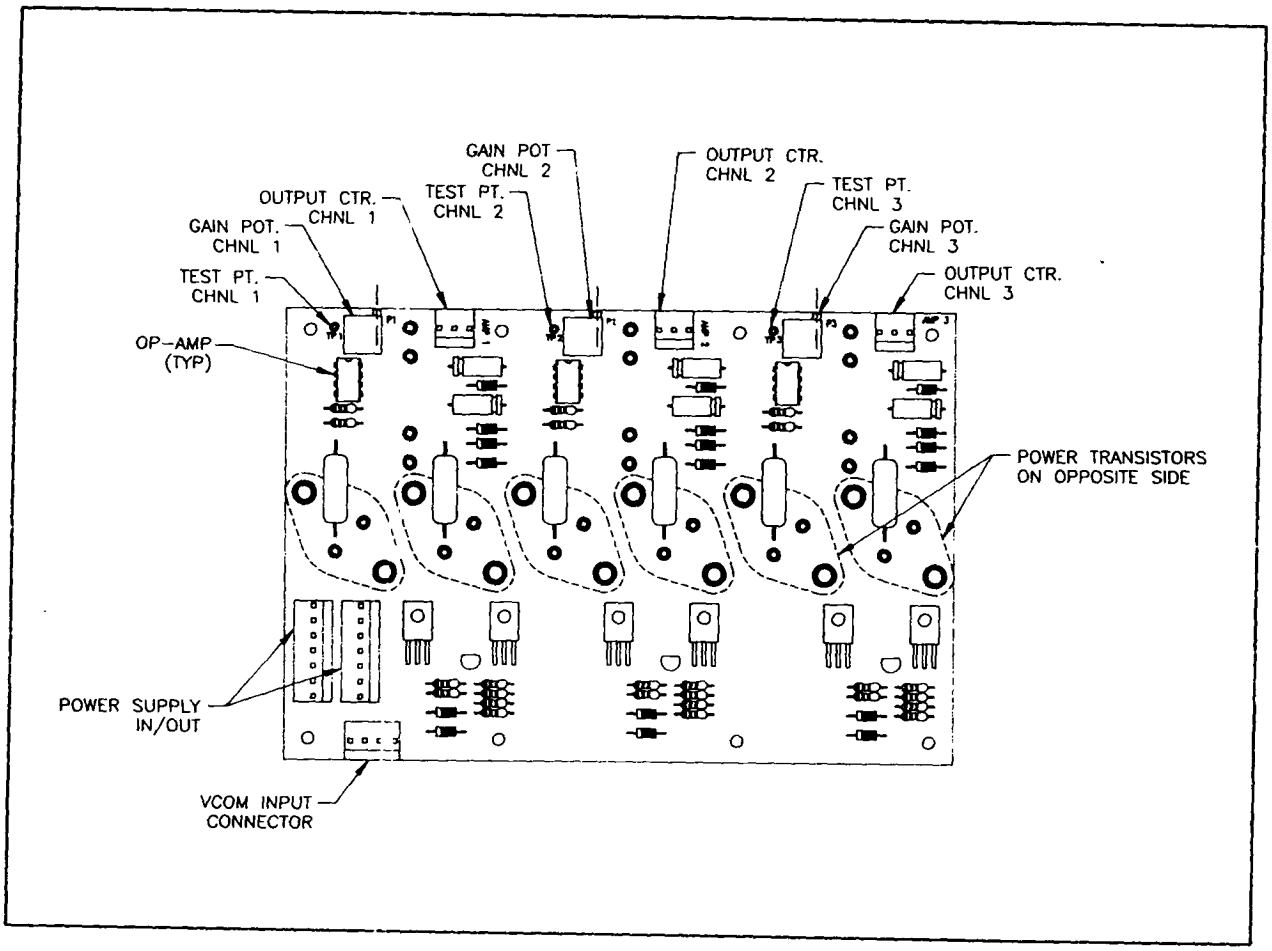


FIGURE 7-3 D.C. Amplifier Module

7-2 MOTHERBOARD (Continued)

Table 7-2 is listed in order from upper left moving clockwise around the board as shown in Figure 7-2.

ITEM	LABEL	FUNCTION
1	J1	Connector for digital I/O numbers 9 to 16.
2	J2	Connector for the arm power relay board.
3	J4	Connector for the teach pendant.
4	J5	Connector for digital I/O numbers 1 to 8.
5	J6	Connector for the auto-start, arm power, and ground.
6	J7	Connector for the gripper.
7	J9	26 pin header for serial I/O channel 1.
8	J10	26 pin header for serial I/O channel 0.
9	J11	Connector for +5VDC, +12VDC, -12VDC, and ground.
10	J12	2 pin connector for the power supply to the encoders.
11	GND	8 pin connector for common ground.
12	J15-22	Eight 6-pin connectors used for the encoder feedback, and velocity command for each axis.
13	RESET	Reset connection from Front panel logic Board.
14	GND	Additional ground connector.

TABLE 7-2 Mother-Board Layout

The Motherboard contains the following:

1) The NEC V30 16-bit microprocessor and the Intel 8087-2 math co-processor running at a master clock frequency of 7.33 MHz.

2) A programable interrupt controller which is used to support the multi-task operating system. It has 8 levels of interrupts which are assigned as follows:

   A150 Series Only:

   - Level 0 is used for the encoders zero crossing signal, used only during homing the robot.
   - Level 1 is used for closing the position loop every 3.8 mSec.
   - Level 2 is used for generating the positional command every 3.8 mSec in the run mode, and 15 mSec. in the manual mode.

   A250 Series Only:

   - Level 0 is used in a poll-mode for communication with the PID axis cards.
   - Level 1 is used for the arm and I/O Watch-dog signals every 3.8 mSec
   - Level 2 is used for generating the positional command to the axis cards. Timing varies depending on mode of operation from 16-40 mSec.

7-4 P-TYPE SERVO AXIS CARDS (A150/151 Only)

The P-Type Servo Axis cards as shown in figure 7-4 contain the necessary functions to close the position loop. It takes an incremental digital command from the mother board, converts it to an analog voltage, and send it to the DC Amplifier. It also takes the feedback signal from the incremental optical encoder in a square wave form (channel A, channel B, and a zero crossing index signal). The signal is then shaped and converted to a pulse train, with pulse width of 1 uSec (+/-30%)

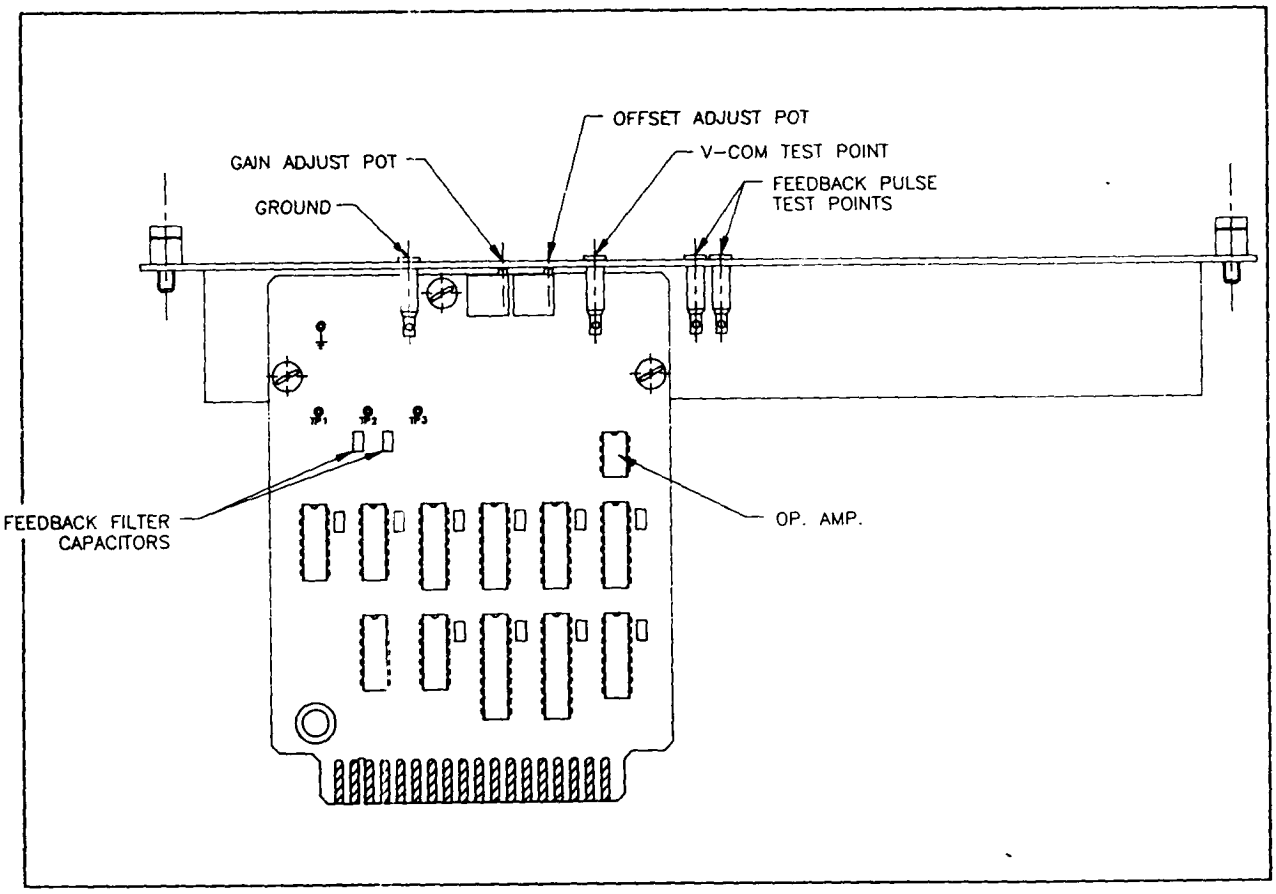


FIGURE 7-4 P-Type Servo Axis Card

7-2 MOTHERBOARD (Continued)

6) Socketed Components

1) U12 - The driver (SN7417) is used to drive the gripper air valve or provide a voltage source for the servo gripper positional feedback. It also supports the ABORT and TEACH push-buttons on the Teach Pendant. If the air valve is shorted, this component is vulnerable to damage.

2) U17 - Serial device receiver (MC1489) converts incoming RS-232 signal levels (+/- 12 Vdc) from both devices 0 and 1 to TTL (0 - 4 Vdc).

3) U18 - Serial device transmitter (MC1488) converts outgoing TTL (0 -4 Vdc) from both devices 0 and 1 to RS-232 signal levels (+/- 12 Vdc).

4) U60 - Power-up Reset and clock generator chip (P8284A) for the V30 CPU.

5) U54 - V30 CPU chip (NEC 70116C).

6) U52 & 58 - RAM Memory Chips. Locations are also used to locate the "piggy-back" RAM expansion board.

7) U19-22 & U30-33 - Locations of the EPROM memory chips (27256 or 27512) which contain the RAPL-II firmware.

The memory section has up to 128 Kbyte of Read Only Memory which contains the RAPL-II executive program. 16 Kbyte of Random Access Memory (Expandable to 64 Kbyte) is provided for the user programing and computer "scratch-pad" space. The RAM chips have an integral lithium energy source which will retain the memory contents for up to 35 years.

It also has eight (8) axis slots, five of which are used for the A100/200 Series robot axes. The other 3 may be used for extra servo axes including a servo gripper. It also has one peripheral slot which contains the local microprocessor bus and is used for the COMBO/32 or EEROM/32 expansion cards.

7-5 PID-TYPE SERVO AXIS CARDS (A250/251 Only)

DESCRIPTION

The PID type axis cards are self-contained micro-controller based servo controllers. A single Intel 8095BH micro-controller chip is used to provide a high performance PID servo loop with a 1 millisecond closure time. The position feedback is updated at the same 1 millisecond intervals. The PID axis card also has the ability to command a trapezoidal or a parabolic velocity profile. The path parameters are generated by the motherboard CPU and are fed to the axis card. A start signal is provided which is then sent to all axis cards in order to provide a synchronized motion.

The motherboard CPU communicates to the PID controller at a variable time base, depending upon the type of operation the robot is performing. When the robot is moving with a joint interpolated motion, the axis card generates all commands, and issues updates of commanded and actual position to the motherboard. When the robot is executing a coordinated path, the motherboard provides command set-points to the axis cards, and then the axis cards perform additional compensation to the commanded signal to provide a smooth output motion.

ANALOG FEEDBACK

Connector 'J2' provides access to the two 10 bit analog to digital converters provided by the 8095BH CPU. These analog inputs can range from 0 to 5 volts.

CONNECTOR J2

PIN	FUNCTION
1	GND
2	ANALOG INPUT CHNL 1
3	ANALOG INPUT CHNL 0

7-3 DC AMPLIFIER MODULE (Continued)

ITEM	FUNCTION

1) 4 pin connector for the command signals connecting the servo axis output to the amplifier module. The signal level is +/- 10 VDC at 70 ma. The pin-out configuration of this connector is:

*	*	*	*
IN1	IN2	IN3	GND.

2) Two 7 pin connectors for the power source are used to connect the amplifier module to the Arm Power Supply carrying +/-27 VDC at 10 amps. The pin-out of this connector is as show below:

*	*	O	*	*	*	*
+ V	+ V	KEY	- V	- V	GND	GND

3) Three 3-pin connectors for the motor power output connecting the amplifier module to the D.C. servo motors. The pin-out is as shown below:

*	*	O
+VM	-VM	KEY

4) Three 20K, 20-turn trim pots. The amplifier gain is increased by turning the adjusting screw CCW.

5) Three test points, TP1, TP2, and TP3. The voltages at the each test point corresponds to the motor armature voltage for the respective channel.

7-5 PID-TYPE SERVO AXIS CARDS (Continued)

FEEDBACK CONFIGURATION

The 'JP2' jumper block is used to determine feedback type, being either single ended (SE) or differential (DI) encoder feedback.

JUMPER BLOCK JP2

PIN	SIGNAL
1	A(SE)
2	A(DI)
3	B(DI)
4	B(SE)
5	Z(SE)
6	Z(DI)

Jumper block 'JP3' is used to determine whether or not the encoder feedback will be used in a multiply x1, x2 or x4 fashion. This facility permits higher effective encoder resolution. For x1 multiplication, no jumpers should be closed. For x4, all should be jumpered.

JUMPER BLOCK JP3

PIN	MULTIPLIER
1	x2, x4
2	x4
3	x4
4	x2, x4
5	x4
6	x4

MOTOR COMMAND SELECTION

Jumper block 'JP4' selects the command voltage output. The axis card generates two types of output. One selection will provide 10 volts of analog output with 12 bit precision. This gives a voltage resolution of 0.0048 volts. The other output selection will provide 0 to 5 volts of pulse-width modulated output. The PWM stage has effective 8 bit resolution, or 0.0195 volts of resolution.

JUMPER BLOCK JP4

PIN	FUNCTION
1	PWM (8 bits)
2	ANALOG (12 bits)

ITEM	FUNCTION
1)	TP1 - the analog command voltage. Should be in the range of +/- 10VDC.
2)	TP2 - the pulse train for the CW motion of the servo motor. The signal at this test point has a pulse width of 1 uSec (+/-30%) and a period depending on the speed of the servo motor.
3)	TP3 - the pulse train for the CCW motion of the servo motor, with the same signal characterstec as TP2.
4)	20K, 20 turn gain adjustment potentiometer. Turning the adjusting screw CCW will increase the gain of the output stage.
5)	20k, 20 turn Offset adjustment potentiometer. Turning the adjusting screw CW will give a positive voltage offset to the output.

TABLE 7-4 P-Type Servo Axis Card Description

7-5 PID-TYPE SERVO AXIS CARDS (Continued)

5) 20k, 20 turn Offset adjustment potentiometer. Turning the adjusting screw CW will give a positive voltage offset to the output.

6) Signal Ground - provides a reference point for measuring the other listed signals.

7) HOME LED - This LED comes on after the axis in question has been homed.

8) FAULT LED - This comes on when the axis is in an error condition. It is reset after recovery.

INPUT/OUTPUT

Header block 'J3' provides access for the TTL level input and output signals that are used by the axis card to detect travel and thermal limits. Outputs are used as status indicators.

CONNECTOR J3

PIN	NAME	FUNCTION
1	VCC	+5 VOLTS
2	IN0	HOME LIMIT
3	IN2	POSITIVE LIMIT
4	IN4	NU
5	IN6	NU
6	OUT0	INTERNAL
7	OUT2	INTERNAL
8	OUT4	READY
9	N/C	
10	N/C	
11	OUT5	HOME
12	OUT3	ERROR
13	OUT1	NU
14	IN7	NU
15	IN5	NU
16	IN3	THERMO LIMIT
17	IN1	NEGATIVE LIMIT
18	GND	SIGNAL GROUND

MOTOR COMMAND/FEEDBACK

Connector 'J50' provides access for all motor command and feedback signals.

CONNECTOR J50

PIN	FUNCTION
1	A
2	A*
3	B
4	B*
5	Z
6	Z*
7	VCOM
8	GND

7-6 FRONT PANEL LOGIC BOARD (Continued)

ITEM	FUNCTION

1) AUTO-START - Connection to the AUTO START switch containing the contact termination and lamp signal.

2) MAIN POWER - Connection to the MAIN POWER switch containing the contact termination and lamp signal.

3) REMOTE FRONT PANEL - Blank connection which carries signals to a remote duplicate front panel.

4) TEACH PENDANT - Connection to the Teach pendant signal cable.

5) ARM POWER BOARD OUTPUT CONNECTOR - Connection to the Arm power relay/filter board (APR/FB). It carries the AC Arm Power (36 VAC centre-tap), 12 Vdc, and Arm enable signal.

6) AC TRANSFORMER CONNECTOR - This connection carries the AC arm power from the secondary of the main transformer the FPLB. This signal is checked by the AC monitoring circuitry on this board and is passed on to the APR/FB.

7) BREAKER SENSOR CONNECTORS - Connection to the motor circuit breakers to detect an overload.

8) DIP SWITCH #1 - 4PST DIP Switch for the following functions:

JP1 - When ON, The remote front panel E-STROP switch must be closed to permit ARM POWER. When OFF, the remote panel E-STOP has no effect.

JP3 - When ON, the AUTO START switch is seen as on at all times. The controller will automatically enter the AUTO_ST program on start-up. When OFF, the AUTO-START switch functions normally.

JP2,4:

These work together as shown below to control the effect of the Emergency stop switch on the TEACH PENDANT. Depending on the application of the robot, it may be an advantage to have the pendant present and have its E-STOP switch active at all times. In other applications, the pendant may not have an E-STOP switch. In still others, the switch should be active only in MANUAL Mode:

7-6 FRONT PANEL LOGIC BOARD

This board contains the arm power switch connections, AC power failure signal, arm power "watch-dog" circuitry, and the Auto Start switch connections. The function of the arm power will only be enabled by the computer if the controller is functioning normally. The continuing safe operation of the computer is checked by the "watch-dog" timer. The 'Arm-Power-On' is a signal to indicate to the mother board the status of the arm power (on/off). The AUTO-START is used in conjunction with the main power switch on the front panel to execute an AUTO_ST program on power-up.

The board connects to the mother board and the I/O termination board via a ribbon cable and a 20-pin edge connector. The latter contains the power supply to the board (+12, +5 VDC and Ground), the ARM-ON signal (to the external DC AMP board), the power fail signal, the AUTO-START signal, and the Arm Enable signal.

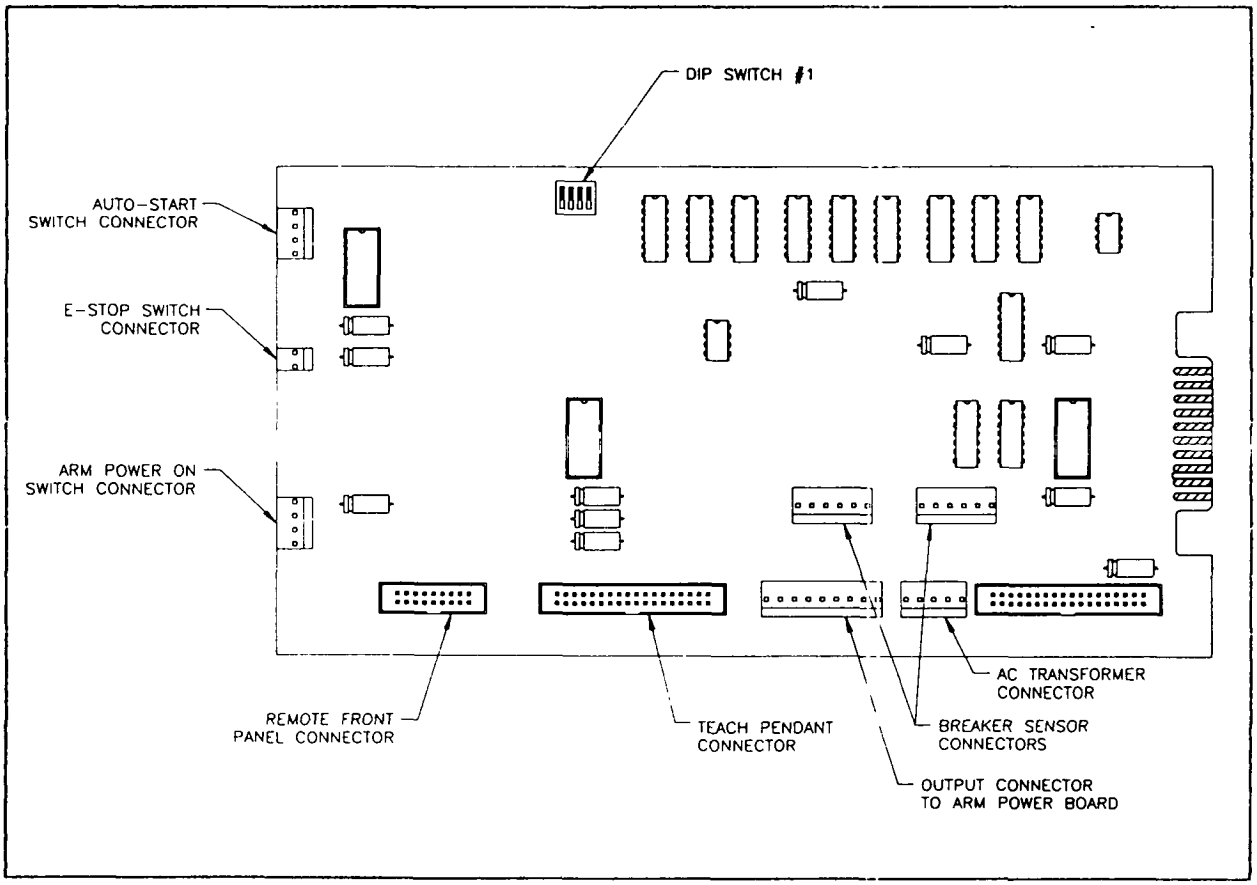


FIGURE 7-6 Front Panel Logic Board

7-7 I/O TERMINATION BOARD

The I/O termination board is shown in figure 7-6. This board has connectors for up to 40 digital inputs and 40 digital outputs at TTL levels un-buffered. It also contains the connectors for the remote E-Stop switch, the auxiliary input (used for example, by the at-home-sensor of the home bracket HOME/RAS) and the analog I/O.

In addition to connectors, it contains the I/O enable watch dog circuitry and the supply to the external buffered I/O racks. This latter supply is fused to protect the circuitry and the enable relay in case of an external short circuit. This fuse is on the board. Its rating is 1 Amp 250 Volt SLOW-BLOW.

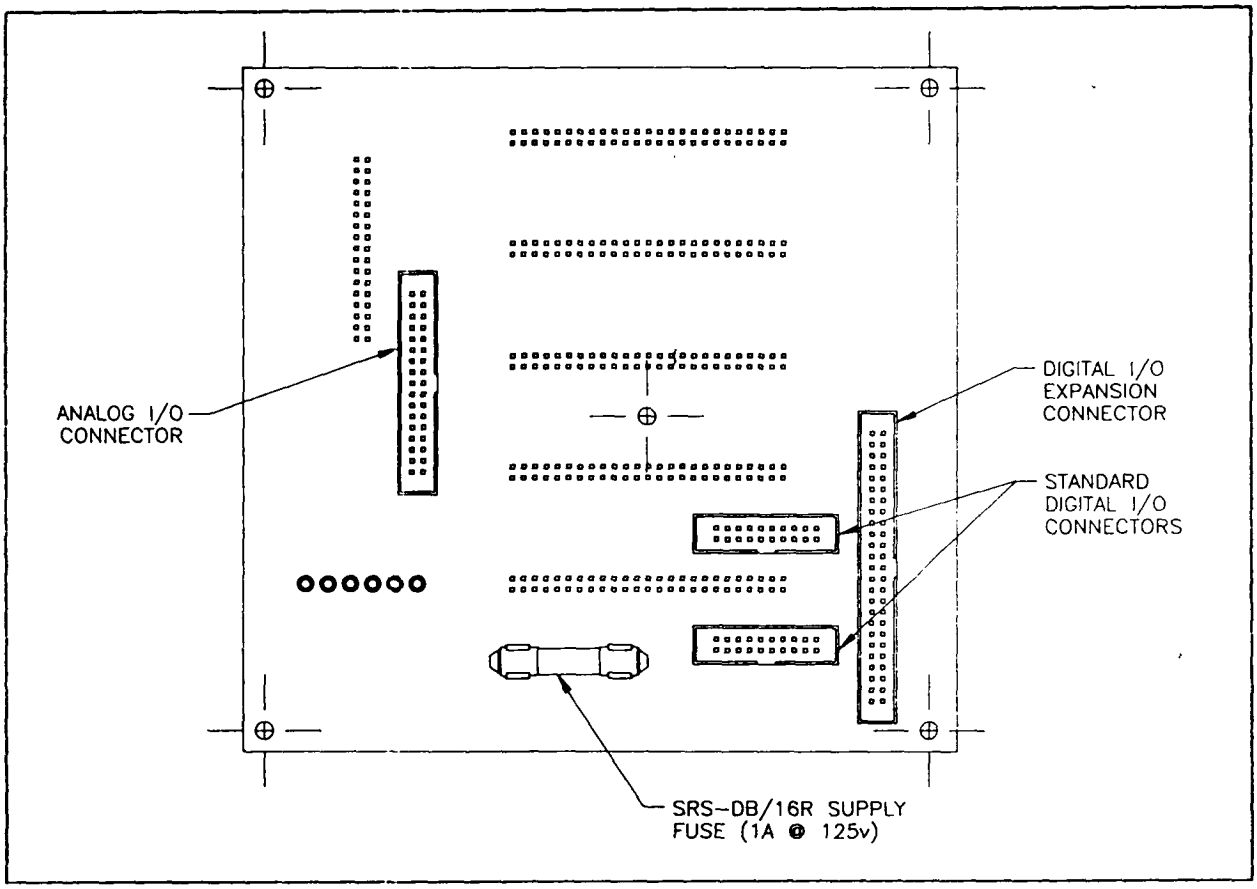


FIGURE 7-7 I/O Termination Board - Viewed from inside controller box.

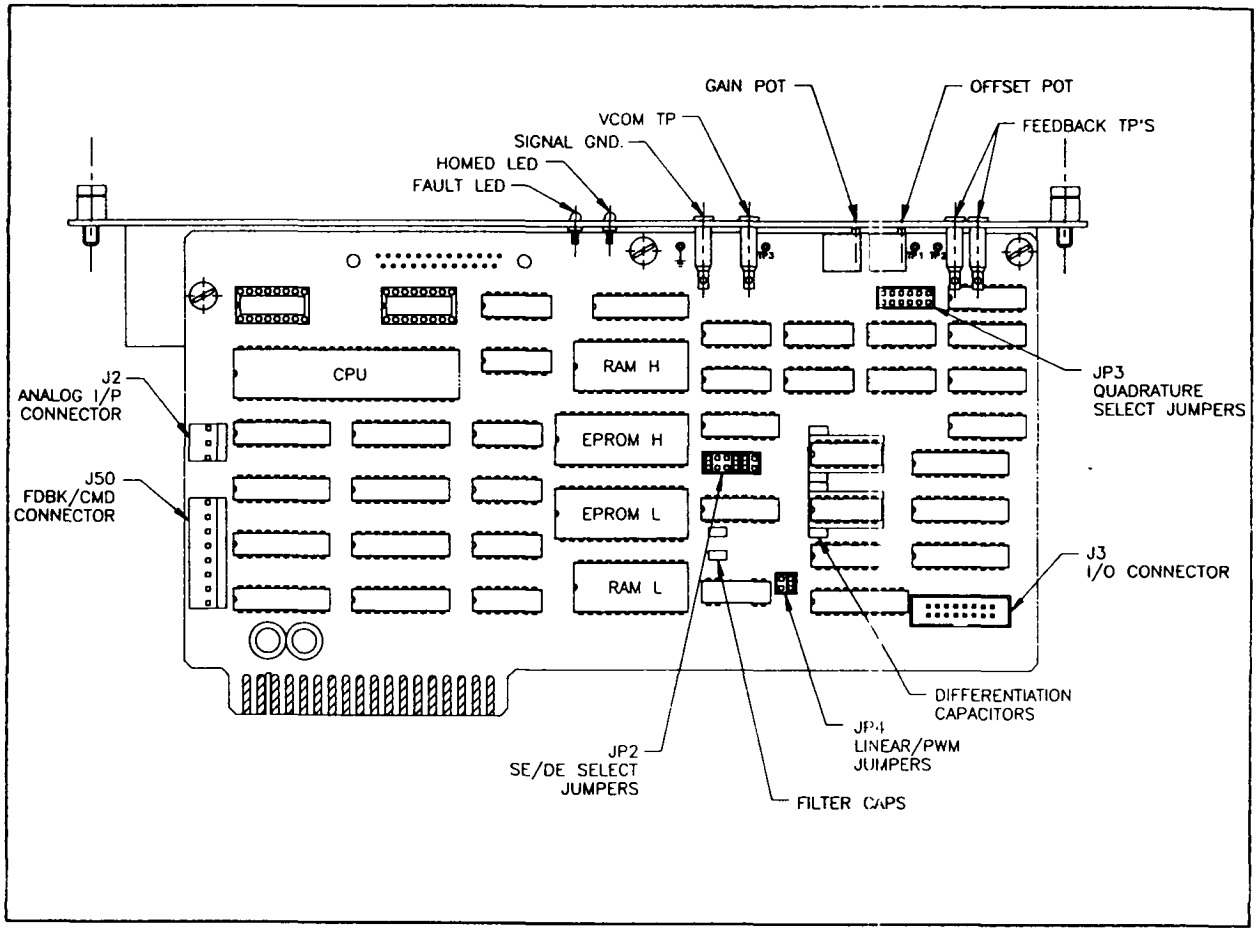


Figure 7-5 PID Type Servo Axis Card

ITEM	FUNCTION

1) TP1 - the analog command voltage. Should be in the range of +/- 10VDC.

2) TP2 - the pulse train for the CW motion of the servo motor. The signal at this test point has a low-true pulse width of 1.8 uSec (+/-30%) and a period depending on the speed of the servo motor.

3) TP3 - the pulse train for the CCW motion of the servo motor, with the same signal characteristic as TP2.

4) 20K, 20 turn gain adjustment potentiometer. Turning the adjusting screw CCW will increase the gain of the output stage.

7-11 ENCODER CONNECTOR BOARD

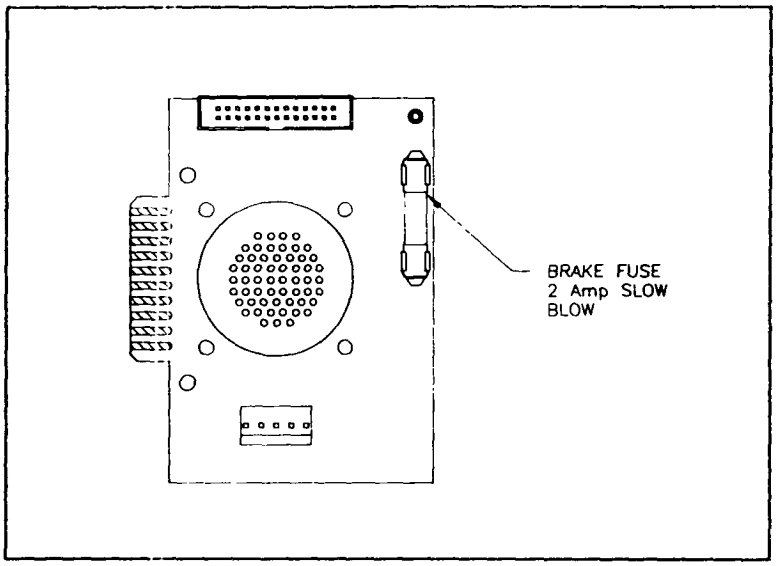


FIGURE 7-8 The Encoder Connector Board.

This board supports the Encoder and Servo Gripper connections to the Robot arm via a 57 pin round connector. It also supports the brake function in the A151/A251 models.

It connects to the Mother board via a 20 pin edge connector for the encoders and servo gripper functions, and a 5 pin connector for the encoder power supply (+5VDC) and ARM ON signal. A 26 pin ribbon cable is used to interconnect this board with the AMP EXPANSION BOARD. The brake fuse is removed on A150/A250 robot systems.

7-8 COMPUTER POWER SUPPLY

The computer power supply is a switching mode supply with three output voltages:

+5 VDC at 8.0 A.
+12 VDC at 2.5 A.
-12 VDC at 1.0 A.

The power supply has output over-voltage clamping, and output short circuit protection.

7-9 ARM POWER SUPPLY

The arm power supply consists of a step_down transformer (36 VAC CT. at 13 A), a bridge rectifier, and two filter capacitors on the Arm Power Filter Board. The output of this power supply is +/- 27 VDC at 10 A, and is unregulated.

7-10 ARM POWER FILTER BOARD

This board takes in rectified AC voltage from the transformer/rectifier via the front panel logic board and filters it into the required +/- 27 Vdc used by the arm power amplifiers. It uses two large filter capacitors to do this. In addition it performs the following three functions:

1) Protect the +/- 27 Vdc ARM POWER SUPPLY by fusing each rail at 10 Amp.

2) Switch the ARM POWER (power to the DC amplifiers) based on a signal from the front panel logic board.

3) Cut off power to both rails of the arm power supply when one of the protection fuses is blown.

8-0 INTRODUCTION

This section of the manual is intended for those that are familiar with Intel 16 bit programming practices. Memory elements defined here are defined as Intel memory items and the definitions of these can be found in the appropriate Intel literature.

System parameters are used to change the robots operating characteristics and to observe what the robot is doing at any particular time.

8-1 MEMORY MANAGEMENT

The CRS Robot System Controller uses the Intel 8086 segmented architecture scheme. The Controller contains three different physical types of memory;

1. Four (4) Kilobytes of Low Power volatile CMOS memory:
   Used for Scratchpad use, 8086 stack space, and the interrupt vector space.

2. Four (4) Kilobytes of Battery-backed CMOS memory:
   Used for System Parameters.

3. Eight (8) Kilobytes of Battery-backed CMOS memory:
   Used for User memory space. This memory can be expanded to 56 Kilobytes.

4. 256 Kilobytes of EPROM memory:
   Used for firmware requirements. This memory can be expanded to 512 Kilobytes for future firmware or custom requirements.

Data integrity is maintained by using checksums on all program, variable, location and path data accesses.

8-2 COMPUTATIONAL ACCURACY

The Robot System Controller software utilizes the Intel 8087 Math co-processor fully for all mathematical calculations. Real numbers are stored in memory as four (4) byte quantities, giving 9 digit precision. Where possible, calculations are made using the full eighty (80) bit (10 byte) capability of the 8087 device, yielding a more accurate result. Although mathematical accuracy may vary, depending upon the nature of the calculations, achievable accuracy will be better than one part per million.

7-12 AMP EXPANSION BOARD

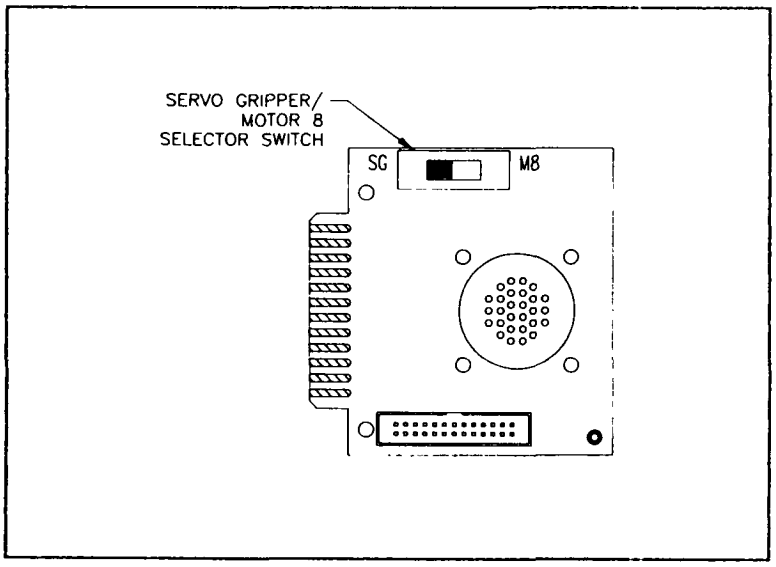


FIGURE 7-9 The Amp Expansion Board

This board supports the differential encoders and command voltages for joints 6, 7, and 8. It connects to the Expansion DC Amplifier via a 28 pin round connector. It uses a 24 pin edge connector to connect to the Mother board, and a 26 conductor ribbon cable to the ENCODER CONNECTOR BOARD.

The SERVO GRIPPER/MOTOR 8 selector switch sets up the function of the mother board axis card #8. When set to SERVO GRIPPER, it routes the gripper position, torque, and motor voltage to/from the arm from/to slot 8. When set to MOTOR 8, it correctly routes the joint 8 encoder and command voltage to/from the expansion DC AMP.

8-4 PATH CONTROL - COMMAND GENERATION

The A150/A250 series of controllers have multiple path generation modes which can be summarized as three major modes of path control:

1) Joint Interpolated motion

MOVE, APPROACH, DEPART, MOTOR, JOINT, MI, MA are RAPL-II commands that create joint interpolated motion. In the A150 controller, the joint interpolated command rate is 4.5 milliseconds. In the A250 controller, the joint interpolated motions are performed at 1 millisecond intervals within the IAXIS axis cards. The Joint Interpolated motion guarantees that all joints to be moved will start and stop together. It also guarantees that maximum acceleration and decceleration will not be exceeded for each axis. Command updates occur at four milli-second intervals. Figure 8-2 shows typical acceleration, velocity and position profiles for a joint interpolated robot motion.

2) Straight line Motion

JOG, straight line MOVE, DEPART, APPROACH, JOG commands provide a straight line path in cartesian space. Cartesian velocity can be controlled by issuing the ENABLE CARTVEL command, which will process the speed command as a percentage of the maximum cartesian velocity (@MAXSPD command). Otherwise, speed control is based on maximum joint velocity. In Straight Line, command updates are performed at a slower rate (18 milliseconds for A150 controller, 16 milliseconds for the A250 controller). This mode of control is a specialized form of the PATH control described below. The "knots" are calculated by RAPL-II based on a linear interpolation between end points. The number of knots used for the straight line is normally 10, which gives reasonable accuracy to the path. If a higher processing speed is required, the number of knots may be reduced. The ROBCOMM package with its memory edit feature can be used for this. Acceleration and decceleration in this mode are based on world coordinates instead of individual joints. If wrist motions are required during the straight line path, then the wrist rotational speeds are continuous throughout the straight line path segment. Joint limit checking is performed before the straight line move, and if any joint exceeds the programmed limits, the command is immediately terminated, and an error message is displayed.

3) Continuous path

CPATH, CTPATH, GOPATH. Defining a path curve through space by selecting a number of intermediate points through which the robot will move. Again, the selection of the CARTVEL parameter will determine the mode of velocity control, but the algorithm always generates a path through the intermediate points.

7-6 FRONT PANEL LOGIC BOARD

This board contains the arm power switch connections, AC power failure signal, arm power "watch-dog" circuitry, and the Auto Start switch connections. The function of the arm power will only be enabled by the computer if the controller is functioning normally. The continuing safe operation of the computer is checked by the "watch-dog" timer. The 'Arm-Power-On' is a signal to indicate to the mother board the status of the arm power (on/off). The AUTO-START is used in conjunction with the main power switch on the front panel to execute an AUTO_ST program on power-up.

The board connects to the mother board and the I/O termination board via a ribbon cable and a 20-pin edge connector. The latter contains the power supply to the board (+12, +5 VDC and Ground), the ARM-ON signal (to the external DC AMP board), the power fail signal, the AUTO-START signal, and the Arm Enable signal.

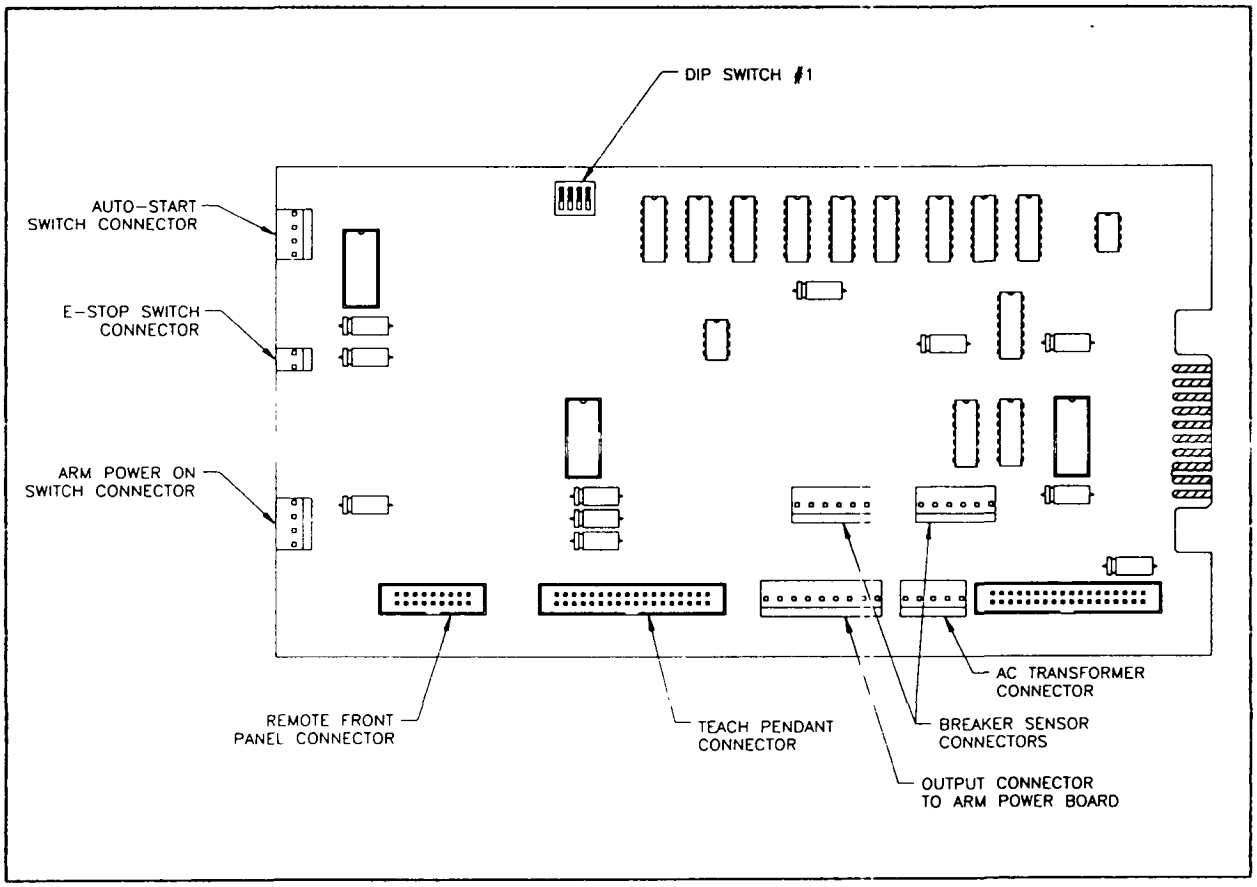


FIGURE 7-6 Front Panel Logic Board

4) VIA path.

The VIA command defines a curve through space based upon a set of intermediate points that will be approximated, but the robot may not move through them, depending upon the acceleration constraints of the robot. The CARTVEL flag determines the mode of velocity control, as before. When CARTVEL is enabled, the path between points is generated as a straight line. With the A150 series, the VIA command is allowed only when CARTVEL is disabled. IA path commands are generated at 16 millisecond intervals for joint interpolated speed profiling, or 32 milliseconds with cartesian velocity profiling selected with the CARTVEL command.

Command generation can be halted using the RAPL-II "HALT ON <[-]Input#>" command and enabling the HOLD parameter. When an interrupt occurs, all joints deccelerate at the rate determined by the characteristics of the filter. When the interrupting signal returns to normal, all joints resume motion also at the rate dictated by the filter stage. This input should be treated as an emergency stop feature, and should not be used as a normal motion hold.

A150 Controller

All command generation output is digitally filtered to provide an update rate of 0.0035 seconds per update for each servo axis.

A250 Controller

Joint interpolated motion is handled entirely by the joint controllers. All other modes of path generation are provided by the motherboard 8086/8087 tandem processors. High levels of motion smoothness is provided by additional compensation by the axis cards, before the commands are sent to the motors. This compensation provides higher performance in terms of speed and smoothness.

JP2	JP4	PENDANT EFFECT ON E-STOP
OFF	OFF	May be unplugged with no effect on Arm Power.
OFF	ON	May be unplugged with no effect on Arm Power.
ON	OFF	Pendant must be present for Arm Power when in Manual mode only.
ON	ON	Pendant must be present for arm power at any time.

8-6 MEMORY ALLOCATION (Continued)

The Program Buffer

The program buffer is a contiguous byte array which contains the user programs. The length of this buffer is set when the ALLOCATE command is entered. Its size depends on what is left of the user memory after allowing for Program, Location and Variable tables, as well as the Reserved memory.

The program buffer stores programs as sequential ASCII characters. Program information is entered by typing in through the user terminal, or downloading from a Master computer using the ACI interface and support software, such as the ROBCOMM software system.

The location of the start of a program is determined by RAPL-II from the index in the program table, relative to the program buffer pointer. Each program is terminated by a dollar-sign character ($) which acts as a seperator. Memory in the program buffer is altered dynamically whenever editing of a program takes place. The memory is "squeezed" up or down as lines are entered, modified, or deleted. Thus the index of all entries in the program table will have to change whenever a line is entered into a program located below the one changed. RAPL-II makes these changes unseen by the programmer. The program buffer can be subdivided into two sections immediately after an ALLOCATE command. The HIMEM command will partition an area away from the uppermost program buffer. This reserved memory will then remain untouched for special programming applications, or PCP's.

Variable Storage

Variables are located in the variable table. The length of this table is defined when the ALLOCATE command is entered. Like the program table, there are three fields for each variable entry in the table. The first is the eight character string used to store the name. The second is a four byte real value which stores the value of the variable. The third is a word quantity used to store the checksum.

Location Storage

Locations (both precision, and cartesian types) are stored in a location table containing ten fields for each location. The first is the 8-character name string. The next eight fields are four-bytes each used to store the components of each location. The RAPL-II controller can control up to 8 individual axes of motion. A precision point can use position information for up to eight axes. In this case, each of the eight fields store a double size integer value of each axis motor coordinate (in encoder pulse units). When a cartesian coordinate is stored, it uses the first six of these 8 fields to store a four byte real number representation of each world component. If a GANTRY command has been specified, the

8-3 PATH CONTROL - CLOSED LOOP CONTROL

The CRS Robot Controller has complete closed loop DC Servo Control for up to eight (8) axes. Gains can be programmable for a wide range of motion requirements. Depending upon the controller type, the servo loop can be closed with different algorithms.

A150 Controller

This is a proportional/differential gain controller. Only the proportional gain is accessible through the RAPL-II GAIN command. The closed loop control is achieved in a period of approximately 3.5 milliseconds. The proportional gain acts on the positional error of each joint of the robot. This error is the difference between the commanded and the actual position of the joint. The actual position is maintained at the servo loop rate. The positional command is updated by the appropriate command generator software, and then is fed through a single order low pass filter with a time constant of 56 milliseconds.

A250 Controller

This controller is based on a fully distributed control architecture, with individual joint controllers. Each joint controller performs a full PID closed loop algorithm in a 1 millisecond loop time, providing high performance. As in the A150 controller, the proportional gain acts upon the difference between the commanded and actual position. The differential gain provides a degree of damping by acting upon changes in joint velocity. The integral gain provides a static error compensation, by acting upon steady state positional errors. The A250 controller is pre-configured at the factory, and only small adjustments to the values should be made when necessary using the RAPL-II GAIN command.

```
char Name[8];
long AxisVal[8];
int  Checksum;
 } *LocationPointer;

CartesianLocation struct
 {
char Name[8];
float Coord[8];
int  Checksum;
 }  *LocationPointer;
 char ProgramBuffer[ProgramBufferSize];
    char *ProgramBufferPointer;
```

The checksums associated with each element of the structures consists of the arithmetic sum of each byte in the structure element. Only the low byte of the checksum is used. The checksum used in the program table is the checksum of the entire program that it references, from the first byte of the program, up to and including the '$' delimiter used to end a program.

Exact addresses for each of the system parameters mentioned above can be found in the memory map for the current version of RAPL-II. This information can be found in Appendix G.

4) VIA path.

The VIA command defines a curve through space based upon a set of intermediate points that will be approximated, but the robot may not move through them, depending upon the acceleration constraints of the robot. The CARTVEL flag determines the mode of velocity control, as before. When CARTVEL is enabled, the path between points is generated as a straight line. With the A150 series, the VIA command is allowed only when CARTVEL is disabled. IA path commands are generated at 16 millisecond intervals for joint interpolated speed profiling, or 32 milliseconds with cartesian velocity profiling selected with the CARTVEL command.

Command generation can be halted using the RAPL-II "HALT ON <[-]Input#>" command and enabling the HOLD parameter. When an interrupt occurs, all joints deccelerate at the rate determined by the characteristics of the filter. When the interrupting signal returns to normal, all joints resume motion also at the rate dictated by the filter stage. This input should be treated as an emergency stop feature, and should not be used as a normal motion hold.

A150 Controller

All command generation output is digitally filtered to provide an update rate of 0.0035 seconds per update for each servo axis.

A250 Controller

Joint interpolated motion is handled entirely by the joint controllers. All other modes of path generation are provided by the motherboard 8086/8087 tandem processors. High levels of motion smoothness is provided by additional compensation by the axis cards, before the commands are sent to the motors. This compensation provides higher performance in terms of speed and smoothness.

9-1 INTRODUCTION

The purpose of the Homing Bracket is to allow simple homing of the robot in a situation where the operator may not be trained for (or be able to) home it accurately using the factory-supplied marks. Such applications are found in industrial systems and labs where the arm may not be easily accessible.

In order to use the homing bracket, the robot calibration must be changed. This is not difficult and will be described in detail in this section of the manual.

9-2 HOMING BRACKET INSTALLATION

The position of the bracket in the robot workspace is important to the success of the procedure. The robot must exit from the bracket using joint or motor commands only. This is because until it is homed, the robot can only make moves which do not depend on the world coordinate system. The most convenient joint to use for exiting is joint 3. The wrist flange must be perpendicular to the upper arm in order to slide cleanly out of the bracket. The angle of the support plate on the bracket is designed to position the bracket as close to the robot as practical and still allow room for typical tooling. The optimum position can be seen in Figure 9-1.

In addition to the bracket itself, the small fixture plate must be installed on the robot wrist flange. Any existing tooling must be removed and the plate installed between the gripper flange and the tooling. It should be installed so that the narrow end would be at the bottom when the robot is at the READY position.

Adding this plate between the robot and the gripper changes the position of the arm in accessing locations in the robot task. The distance can be compensated by re-teaching the locations or by adding a tool offset of 0.187 in the X direction. If no tool offset is used already, enter a location NEWTOOL with the value ".187,0,0,0,0,0". Then issue the command "TOOL NEWTOOL". Re-teaching the locations is usually a good idea any time the gripper comes off the robot, as it may not be repositioned exactly where it was before.

8-4 PATH CONTROL - COMMAND GENERATION

The A150/A250 series of controllers have multiple path generation modes which can be summarized as three major modes of path control:

1) Joint Interpolated motion

MOVE, APPROACH, DEPART, MOTOR, JOINT, MI, MA are RAPL-II commands that create joint interpolated motion. In the A150 controller, the joint interpolated command rate is 4.5 milliseconds. In the A250 controller, the joint interpolated motions are performed at 1 millisecond intervals within the IAXIS axis cards. The Joint Interpolated motion guarantees that all joints to be moved will start and stop together. It also guarantees that maximum acceleration and decceleration will not be exceeded for each axis. Command updates occur at four milli-second intervals. Figure 8-2 shows typical acceleration, velocity and position profiles for a joint interpolated robot motion.

2) Straight line Motion

JOG, straight line MOVE, DEPART, APPROACH, JOG commands provide a straight line path in cartesian space. Cartesian velocity can be controlled by issuing the ENABLE CARTVEL command, which will process the speed command as a percentage of the maximum cartesian velocity (@MAXSPD command). Otherwise, speed control is based on maximum joint velocity. In Straight Line, command updates are performed at a slower rate (18 milliseconds for A150 controller, 16 milliseconds for the A250 controller). This mode of control is a specialized form of the PATH control described below. The "knots" are calculated by RAPL-II based on a linear interpolation between end points. The number of knots used for the straight line is normally 10, which gives reasonable accuracy to the path. If a higher processing speed is required, the number of knots may be reduced. The ROBCOMM package with its memory edit feature can be used for this. Acceleration and deceleration in this mode are based on world coordinates instead of individual joints. If wrist motions are required during the straight line path, then the wrist rotational speeds are continuous throughout the straight line path segment. Joint limit checking is performed before the straight line move, and if any joint exceeds the programmed limits, the command is immediately terminated, and an error message is displayed.

3) Continuous path

CPATH, CTPATH, GOPATH. Defining a path curve through space by selecting a number of intermediate points through which the robot will move. Again, the selection of the CARTVEL parameter will determine the mode of velocity control, but the algorithm always generates a path through the intermediate points.

having to measure the true zero location as described in the Technical Manual, page A-5.

The robot must then be moved into the home bracket. This is most easily done with the ARM POWER off. When the arm is in position, turn arm power back on. set the speed to a slow value like 10%. Issue a joint command to move the robot out of the bracket such as "JOINT 3,20". Watch the arm carefully to make sure it leaves the bracket easily without binding. Application of a bit of grease to the inner edge of the bracket can ease the robot's exit.

Once the robot has exited the bracket, it must have space to perform the Homing sequence during which it moves slightly. The position arrived at after the 20 degree joint 3 move may not be acceptable due to interference with external equipment etc. In this case, use further JOINT commands to move the arm to a clear location. Record the sequence of moves taken to get clear as the robot will have to repeat them each time it is homed. Once at a clear location, use the @@CAL command to recalibrate the robot at the new home location.

There is one note of caution to be added. If during the calibration sequence, an individual move of a joint to get to the encoder index mark takes very little time to complete, that joint should be commanded to move back by about 1/2 a motor turn (200 to 500 pulses) to ensure that it sees the correct index mark every time.

In addition, if the joint takes almost a complete turn of the motor to complete, the joint should be compensaated forward by about 200 to 500 pulses. Such a correction should be used in the homing program (see below).

having to measure the true zero location as described in the Technical Manual, page A-5.

The robot must then be moved into the home bracket. This is most easily done with the ARM POWER off. When the arm is in position, turn arm power back on. set the speed to a slow value like 10%. Issue a joint command to move the robot out of the bracket such as "JOINT 3,20". Watch the arm carefully to make sure it leaves the bracket easily without binding. Application of a bit of grease to the inner edge of the bracket can ease the robot's exit.

Once the robot has exited the bracket, it must have space to perform the Homing sequence during which it moves slightly. The position arrived at after the 20 degree joint 3 move may not be acceptable due to interference with external equipment etc. In this case, use further JOINT commands to move the arm to a clear location. Record the sequence of moves taken to get clear as the robot will have to repeat them each time it is homed. Once at a clear location, use the @@CAL command to recalibrate the robot at the new home location.

There is one note of caution to be added. If during the calibration sequence, an individual move of a joint to get to the encoder index mark takes very little time to complete, that joint should be commanded to move back by about 1/2 a motor turn (200 to 500 pulses) to ensure that it sees the correct index mark every time.

In addition, if the joint takes almost a complete turn of the motor to complete, the joint should be compensaated forward by about 200 to 500 pulses. Such a correction should be used in the homing program (see below).

9-4 AUTO START PROGRAM (Continued)

The E_STOP subroutine is called whenever a safe condition is breached in the workspace, or when the normal end of the job occurs. This routine first moves the robot to a safe location (SAFE), The IGNORE command causes the robot to ignore the stop signal which caused it to jump to this routine. It then moves back into the homing bracket, and the ARM POWER is turned off.

```
PROGRAM E_STOP
10 MOVE SAFE
20 IGNORE
30 MOVE #ATCAL
40 MOVE #PREHOME
50 SPEED 10
60 MOVE #FIXTURE
70 FINISH
75 DELAY 1
80 DISABLE ARM
90 STOP
$
```

In this program, the location #ATCAL is the position of the arm where the homing sequence is called. Using a precision point in all these locations is recommended here as changing the TOOL transform (if say the tool length changes slightly) will re-position the arm and may cause a collision with the bracket.

#PREHOME is a location just above the bracket itself, and #FIXTURE is the position of the robot in the homing bracket.

Notice that turning the arm off is permitted in a program, however an error will result, stopping the program. This has no effect on the operation of this program.

9-1 INTRODUCTION

The purpose of the Homing Bracket is to allow simple homing of the robot in a situation where the operator may not be trained for (or be able to) home it accurately using the factory-supplied marks. Such applications are found in industrial systems and labs where the arm may not be easily accessible.

In order to use the homing bracket, the robot calibration must be changed. This is not difficult and will be described in detail in this section of the manual.

9-2 HOMING BRACKET INSTALLATION

The position of the bracket in the robot workspace is important to the success of the procedure. The robot must exit from the bracket using joint or motor commands only. This is because until it is homed, the robot can only make moves which do not depend on the world coordinate system. The most convenient joint to use for exiting is joint 3. The wrist flange must be perpendicular to the upper arm in order to slide cleanly out of the bracket. The angle of the support plate on the bracket is designed to position the bracket as close to the robot as practical and still allow room for typical tooling. The optimum position can be seen in Figure 9-1.

In addition to the bracket itself, the small fixture plate must be installed on the robot wrist flange. Any existing tooling must be removed and the plate installed between the gripper flange and the tooling. It should be installed so that the narrow end would be at the bottom when the robot is at the READY position.

Adding this plate between the robot and the gripper changes the position of the arm in accessing locations in the robot task. The distance can be compensated by re-teaching the locations or by adding a tool offset of 0.187 in the X direction. If no tool offset is used already, enter a location NEWTOOL with the value ".187,0,0,0,0,0". Then issue the command "TOOL NEWTOOL". Re-teaching the locations is usually a good idea any time the gripper comes off the robot, as it may not be repositioned exactly where it was before.

CONTENTS

```
char Name[8];
long AxisVal[8];
int  Checksum;
 } *LocationPointer;

CartesianLocation struct
 {
char Name[8];
float Coord[8];
int  Checksum;
 }  *LocationPointer;
 char ProgramBuffer[ProgramBufferSize];
    char *ProgramBufferPointer;
```

The checksums associated with each element of the structures consists of the arithmetic sum of each byte in the structure element. Only the low byte of the checksum is used. The checksum used in the program table is the checksum of the entire program that it references, from the first byte of the program, up to and including the '$' delimiter used to end a program.

Exact addresses for each of the system parameters mentioned above can be found in the memory map for the current version of RAPL-II. This information can be found in Appendix G.

@ACCEL

TOKEN: /013

FORMAT:

@ACCEL (axis#),<value>

DESCRIPTION:

Alter the maximum acceleration rate parameter for any axis. This value can be (0 < value <= 2) and is a real number.

The unit of <value> is in pulses per tick/tick, where a tick is the minimum unit of time used by the command generation software and a pulses is the basic length unit of the servo.

For the A150 robot with standard axis cards, the default value is 1 ie. 1/1000 of a motor turn per 0.004^2 seconds^2. This gives a maximum acceleration of 62.5 revolutions/sec^2.

For the A250 controller the time base for this calculation is much shorter. The default value is 0.0493 1/1000ths of a motor rev. per $.001^2$ seconds^2. This gives an acceleration of 49.3 revolutions/sec^2. The system should not exceed 120 revolutions/sec^2 which is .12 units.

APPLICABLE MODES:

{I},{M}

8-6 MEMORY ALLOCATION (Continued)

The Program Buffer

The program buffer is a contiguous byte array which contains the user programs. The length of this buffer is set when the ALLOCATE command is entered. Its size depends on what is left of the user memory after allowing for Program, Location and Variable tables, as well as the Reserved memory.

The program buffer stores programs as sequential ASCII characters. Program information is entered by typing in through the user terminal, or downloading from a Master computer using the ACI interface and support software, such as the ROBCOMM software system.

The location of the start of a program is determined by RAPL-II from the index in the program table, relative to the program buffer pointer. Each program is terminated by a dollar-sign character ($) which acts as a seperator. Memory in the program buffer is altered dynamically whenever editing of a program takes place. The memory is "squeezed" up or down as lines are entered, modified, or deleted. Thus the index of all entries in the program table will have to change whenever a line is entered into a program located below the one changed. RAPL-II makes these changes unseen by the programmer. The program buffer can be subdivided into two sections immediately after an ALLOCATE command. The HIMEM command will partition an area away from the uppermost program buffer. This reserved memory will then remain untouched for special programming applications, or PCP's.

Variable Storage

Variables are located in the variable table. The length of this table is defined when the ALLOCATE command is entered. Like the program table, there are three fields for each variable entry in the table. The first is the eight character string used to store the name. The second is a four byte real value which stores the value of the variable. The third is a word quantity used to store the checksum.

Location Storage

Locations (both precision, and cartesian types) are stored in a location table containing ten fields for each location. The first is the 8-character name string. The next eight fields are four-bytes each used to store the components of each location. The RAPL-II controller can control up to 8 individual axes of motion. A precision point can use position information for up to eight axes. In this case, each of the eight fields store a double size integer value of each axis motor coordinate (in encoder pulse units). When a cartesian coordinate is stored, it uses the first six of these 8 fields to store a four byte real number representation of each world component. If a GANTRY command has been specified, the

TOKEN: /019

FORMAT:

@@CAL

DESCRIPTION:

Factory calibrate command used at Factory. Do **NOT** use this command unless servicing the robot.

WARNING

DO NOT USE THIS COMMAND, UNLESS ON THE ADVICE OF CRS PLUS.

EXAMPLE

Step	Action
1	Do a "TEACH" start.
2	Arm power should be on.
3	Robot should be in "MANUAL" mode.
4	Move arm to zero position as shown in figure A-1.
5	Enter "NOMANUAL".
6	Enter "PASSWORD".
7	Support arm so it doesn't drop during zeroing.
8	Enter "@ZERO".
9	Enter "JOINT 2,90"
10	Check each joint to ensure they are in "HOME" range. If not repeat.
11	Enter "@@CAL Y".

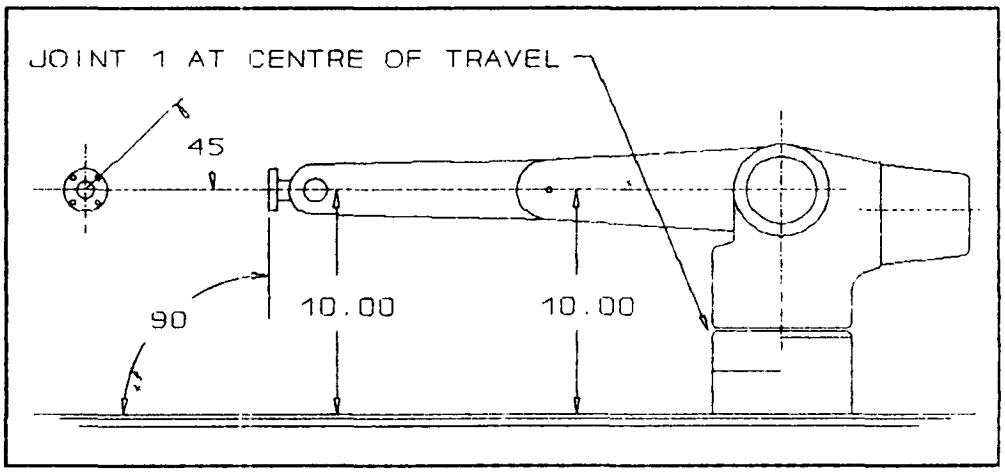


Figure A-1 Pose of the arm at true "zero" (CALRDY Pose)

TOKEN: /

FORMAT:

@@DIAG

DESCRIPTION:

This command permits the system programmer to interrogate the operation of the controller. This should be done when the controller is considered to have some form of failure.

WARNINGS:

1. After diagnostics have been run, the controller should be turned OFF, then back ON again. This will clear any pending diagnostic operations and will return the controller to the state of readiness.

2. The diagnostic procedure will request that the arm power be turned off before the diagnostic procedures begin. **FAILURE TO TURN THE ARM POWER OFF MAY CAUSE UNEXPECTED ARM MOTIONS AND POTENTIAL DAMAGE TO PEOPLE AND MACHINES COULD RESULT.**

3. The diagnostic test procedure should be exercised by, or under the guidance of, trained service personnel.

TOKEN: /116

FORMAT:

@DIGIO <MODE>

DESCRIPTION:

This command enables the system programmer to select the number of physical inputs and outputs that are available for system integration. This function is useful only when the COMBO/32 option is installed, since the basic 16 inputs and outputs are not programmable.

The MODE argument is a number from 1 to 8 inclusive that determines the operating characteristic of all 6 banks of programmable I/O. To indicate the I/O bank number and its mode of operation, the table uses the letter 'O' for Output and 'I' for Input followed by a number. The number indicates the bank number. The standard I/O uses bank 0 and 1 (there are 16 standard inputs and outputs). Each bank has 8 inputs and 8 outputs. Consequently, the programmable I/O on the combo card starts at bank 2. The default condition of the combo card I/O is mode #0. The table below indicates the operating mode of all of the combo-card banks with any given mode input:

This translates to which cable connection at the controller will provide the proper signals. Since each connector handles 16 I/O points, there are three groups of 16 I/O. The letters A, B and C correspond to the connectors on the back panel labelled 17-24, 25-32, 33-40 respectively. These connector designations will no longer make sense if any other mode but 1 or 5 is programmed.

Mode	Port ID					
	A0	B0	C0	A1	B1	C1
1	O2	O3	O4	I2	I3	I4
2	I9	I8	I7	I2	I3	I4
3	O2	I8	I7	I2	I3	I4
4	O2	O3	I7	I2	I3	I4
5	O2	O3	O4	I2	I3	I4
6	O2	O3	O4	I2	I3	O5
7	O2	O3	O4	I2	O6	O5
8	O2	O3	O4	O7	O6	O5

TABLE A-3 Default I/O modes after Teach Start

Assuming that the diagnostic mode has been entered successfully, a variety of options are available. The options are selected according to the letter provided. The following list of options is displayed:

```
ENTER ONE OF THE SELECTIONS BELOW:
A = PIC R/W TEST
B = PIT COUNTING TEST
C = PIT INTERRUPT CAPABILITY TEST
D = 8087 MATH CO-PROCESSOR TEST
E = TEACH PENDANT I/O TEST
F = ANALOG INPUT TEST
G = ANALOG OUTPUT TEST
H = DIGITAL INPUT TEST
I = DIGITAL OUTPUT TEST
J = AXIS CARD TEST
K = MEMORY TEST
L = RTC TEST
M = MONITOR
N = EEPROM TEST
O = ALL TESTS
P = EXIT DIAGNOSTICS
SELECTION?
```

SELECTION A - Programmable Interrupt Controller Test

When this option is selected, the screen displays:

```
PIC R/W TEST
```

This test provides data bus and chip addressability requirements. Status words are read and written to the 8259 chip. Results of the test will show PASS or FAIL. Any read/write failure will be shown as:

```
PIC R/W FAIL WRITTEN ww:
READ: rr
```

Where ww and rr are byte values written to and read from the device.

SELECTION B - Programmable Interval Timer Test

When this option is selected, the screen displays:

```
PIT COUNTING TEST
```

Similar to test A, this test provides data bus and chip addressability requirements. Status words are read and written to the 8254 chip. Timer values are written into the chip, and elapsed counter values are compared to nominal

When selection 'A' is made, the system prompts the user to enter the current time and date information. Time is entered in 24 hour military format. The 'DAY' code is entered as a number from 1 to 7 representing Monday through Sunday. Once entered, the chip is set and is removed from its internal low power consumption mode where the internal oscillator is not active. The timer circuit is activated from this point on.

Selection 'B' de-activates the timer circuit and the clock stops. This useful only before removing the chip and placing it into its internal low-consumption mode for storage.

Selection 'C' displays the timer contents repetitively. If the seconds register fails to change with time, then repeat step 'A'. Continuing failure of this test indicates a failed timer chip, or the chip does not reside in the proper socket.

Selection 'D' returns to the main diagnostic level.

SELECTION M - Monitor Commands

The monitor is a low level control language which can be used to check, change and run the contents of the controller memory. Five one-letter monitor commands are available in this mode. The menu which appears when 'M' is selected lists these commands:

```
"S"UBSTITUTE
"D"UMP
"I"NPUT
"O"UTPUT
"G"O
"E"XIT
MON>>
```

The "MON>>" prompt will re-appear after each command is executed. When "E" is pressed, the main menu will re-appear.

The Substitute and Dump commands use the data types shown in Table A1.

The address of the data to be examined or modified must be entered in standard Intel Format of segment:offset.

In the Substitute command, once the data has been entered, a <CR> will terminate the command, but a <SPACE> will prompt the next data location to be changed. Never enter a blank line of data, as it will be read as "0".

The Dump command prompts for the number of data elements to be displayed. Both the DUMP and the SUBSTITUTE commands display and expect data in decimal format.

The INPUT and OUTPUT commands read and write hexadecimal data values to and from the 8086 physical I/O space. Note that unlike the DUMP or SUBSTITUTE commands, hexadecimal values are displayed.

SELECTION E - Teach Pendant Input/Output Test

When this option is selected, the screen displays:

TEACH PENDANT TEST

This test procedure leads the user through several different inputs and outputs that are part of teach pendant function. Each pendant switch is tested by the user with the result being displayed. The potentiometer feedback value is displayed, and the READY lamp is flashed to indicate correct operation. In addition, the AUTO-START switch is tested. Each test is headed by a message prompting the user for the correct action.

a) READY LAMP!! - "E" to end

The READY lamp will flash until the user enters the E character, or until the test times out, which is approximately 1 minute.

b) GRIPPER OUTPUT!! - "E" to end

During this section of the diagnostics, the pneumatic gripper solenoid will be tested. On A150/A250 robots, the gripper solenoid can be heard easily, verifying correct operation. On other systems, the voltage must be measured at the proper test point to verify correct operation.

Input states - "E" to end

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b

The direction switch inputs are all scanned. All bits ("b" above) will appear as 1 or 0 as a result. Normally, the inputs will show a 1, or high value. The assignment of these inputs are described in the technical manual, and in the service manual.

c) Press AUTOSTART

Although not a teach pendant function, the diagnostic test waits until the front panel auto start switch has been toggled.

d) Press TEACH

The system waits until the TEACH push button has been pressed and released.

e) Press ABORT

The system waits until the ABORT push button has been pressed and released.

Test 6 - Communication Diagnostic Test

This procedure will test the integrity of the axis card to motherboard communication. The test communicates to one or all axis cards so that data is read and written to ensure data accuracy. The user can execute this test with all axis or one by responding to the prompt:

```
ENTER AXIS CARD NUMBER FOR TEST (0 FOR ALL):N (N is 0 or 1 to 8)
```

The following display appears:

```
Any Key to end...
Ax Good  Bad
#  Cycles     Cycles
N  GGGGGGGGG  BBBBBBBBB
```

If a single axis is selected, then the number of good and bad communication cycles will be constantly updated. If all cards are selected, then the display will remain blank until a key is pressed to terminate the test. At that time, the results of all axes will be displayed.

```
Any Key to end...
Ax Good  Bad
#  Cycles     Cycles
1  GGGGGGGGG  BBBBBBBBB
2  GGGGGGGGG  BBBBBBBBB
3  GGGGGGGGG  BBBBBBBBB
4  GGGGGGGGG  BBBBBBBBB
5  GGGGGGGGG  BBBBBBBBB
6  GGGGGGGGG  BBBBBBBBB
7  GGGGGGGGG  BBBBBBBBB
8  GGGGGGGGG  BBBBBBBBB
```

The cycle test counters can count up to 2 billion test cycles, which typically would be about 200 days of continuous testing.

If a communication problem exists with the axis card, ie. all of the communication control lines do not indicate a healthy communication, then the following will appear:

```
********* Bad Communication *********
```

a) Test 1 will provide a fast check of the analog outputs. The teach pendant potentiometer is used as an input for the value which is output to the converters. To test both converters simultaneously, channel 1 will always output the value that has been input to it and channel 2 will display the complimentary output of channel 1. In this way, a 2-channel scope can be applied to both outputs channels in differential mode. The result should always be constant If not, a bit may be stuck in the analog circuitry.

```
Analog output #1 will reflect true output always
Analog output #2 will be 1's compliment of output #1 always - Enter 7
```

b) Test 2 outputs a sawtooth wave to the two analog output channels. These can be viewed and checked with a scope.

c) Test 3 displays the following message:

```
Enter decimal byte value:
```

The value entered at this prompt is sent to the two converters. A scope or DVM can be used to check for the correct value. If either of the first two tests indicate a problem, this test can be used to look for the exact location of the failure. Incrementing through all possible values (0 to 255) should show a smooth progression in analog value. Any step changes should be taken as a failure.

SELECTION H - Digital Input Test

This test displays the state of all digital inputs. The function of the digital input sections is described by the header line. For more complete information regarding the digital input usage, consult the Technical Manual. Inputs will be displayed as either 1 or 0.

```
Test for all digital Inputs
Hit any key to end
<TP Axis Switch><TP i/p><BDPO-7><Program 1-16 ><17--24><25--32><33--40>
0     10          20                    30                    4050
|     |           |                     |                     ||
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
```

Press any key to end this continuous scanning test.

SELECTION I - Digital Output Test

This procedure tests the digital outputs of the controller. When this test is initiated, the digital outputs are all toggled at a high rate. The outputs should appear on a scope as a square wave. Adjacent outputs will be inverted. In this way, outputs that are tied together will be obvious.

```
Test for all digital Outputs
Outputs will toggle, alternate outputs out of phase - Hit any key to end
```

A250 controller

The A250 controller has more comprehensive tests to establish whether or not the smart axis cards are functioning correctly. When the axis test is selected, a new menu is displayed:

```
TESTING AXIS CARD #N   N = 1 TO 8
ENTER ONE OF THE FOLLOWING TEST CODES:
1 - WRITE TO AXIS CARD,
2 - READ FROM AXIS CARD
3 - SEND RESET SIGNAL TO AXIS CARD
4 - READ FULL AXIS CARD STATUS
5 - READ ERROR CODES FROM AXIS CARD
6 - COMMUNICATION DIAGNOSTIC TEST
7 - SELECT NEW AXIS CARD FOR TEST
8 - END ALL TESTS
REQUEST:
```

This menu is displayed after each test. It will always show the current axis card being tested. Entering #8 will retrun to the main diagnostic menu.

NOTE: Options 1 and 2 assume that the user has prior knowledge of the axis card command set, and the structure of the communication function. This information is found in the technical manual.

Test 1 - Axis Card Write

This Test permits the user to enter a command code and data into the axis card. After selecting this option, the prompt:

```
ENTER HEX CODE:
```

will appear. The user must then enter a correct command/data code in hexadecimal format. The system will respond with:

```
**PASSED      or      **FAILED
```

This indicates the success of the transfer.

Test 2 - Axis Card Read

This test reads back the next available data word from the axis card. This command assumes that data should be available. If it is not available, then a **FAILED message is displayed, otherwise the following line will appear:

```
HHHH  **PASSED
```

where HHHH is the hex code of the returned data word.

A250 controller

The A250 controller has more comprehensive tests to establish whether or not the smart axis cards are functioning correctly. When the axis test is selected, a new menu is displayed:

```
TESTING AXIS CARD #N    N = 1 TO 8
ENTER ONE OF THE FOLLOWING TEST CODES:
1 - WRITE TO AXIS CARD,
2 - READ FROM AXIS CARD
3 - SEND RESET SIGNAL TO AXIS CARD
4 - READ FULL AXIS CARD STATUS
5 - READ ERROR CODES FROM AXIS CARD
6 - COMMUNICATION DIAGNOSTIC TEST
7 - SELECT NEW AXIS CARD FOR TEST
8 - END ALL TESTS
REQUEST:
```

This menu is displayed after each test. It will always show the current axis card being tested. Entering #8 will retrun to the main diagnostic menu.

NOTE: Options 1 and 2 assume that the user has prior knowledge of the axis card command set, and the structure of the communication function. This information is found in the technical manual.

Test 1 - Axis Card Write

This Test permits the user to enter a command code and data into the axis card. After selecting this option, the prompt:

```
ENTER HEX CODE:
```

will appear. The user must then enter a correct command/data code in hexadecimal format. The system will respond with:

```
**PASSED      or      **FAILED
```

This indicates the success of the transfer.

Test 2 - Axis Card Read

This test reads back the next available data word from the axis card. This command assumes that data should be available. If it is not available, then a **FAILED message is displayed, otherwise the following line will appear:

```
HHHH  **PASSED
```

where HHHH is the hex code of the returned data word.

a) Test 1 will provide a fast check of the analog outputs. The teach pendant potentiometer is used as an input for the value which is output to the converters. To test both converters simultaneously, channel 1 will always output the value that has been input to it and channel 2 will display the complimentary output of channel 1. In this way, a 2-channel scope can be applied to both outputs channels in differential mode. The result should always be constant. If not, a bit may be stuck in the analog circuitry.

```
Analog output #1 will reflect true output always
Analog output #2 will be 1's compliment of output #1 always - Enter 7
```

b) Test 2 outputs a sawtooth wave to the two analog output channels. These can be viewed and checked with a scope.

c) Test 3 displays the following message:

```
Enter decimal byte value:
```

The value entered at this prompt is sent to the two converters. A scope or DVM can be used to check for the correct value. If either of the first two tests indicate a problem, this test can be used to look for the exact location of the failure. Incrementing through all possible values (0 to 255) should show a smooth progression in analog value. Any step changes should be taken as a failure.

SELECTION H - Digital Input Test

This test displays the state of all digital inputs. The function of the digital input sections is described by the header line. For more complete information regarding the digital input usage, consult the Technical Manual. Inputs will be displayed as either 1 or 0.

```
Test for all digital Inputs
Hit any key to end
<TP Axis Switch><TP i/p><BDPO-7><Program 1-16 ><17--24><25--32><33--40>
0     10          20                  30                  4050
|     |           |                   |                   ||
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
```

Press any key to end this continuous scanning test.

SELECTION I - Digital Output Test

This procedure tests the digital outputs of the controller. When this test is initiated, the digital outputs are all toggled at a high rate. The outputs should appear on a scope as a square wave. Adjacent outputs will be inverted. In this way, outputs that are tied together will be obvious.

```
Test for all digital Outputs
Outputs will toggle, alternate outputs out of phase - Hit any key to end
```

Test 6 - Communication Diagnostic Test

This procedure will test the integrity of the axis card to motherboard communication. The test communicates to one or all axis cards so that data is read and written to ensure data accuracy. The user can execute this test with all axis or one by responding to the prompt:

```
ENTER AXIS CARD NUMBER FOR TEST (0 FOR ALL):N (N is 0 or 1 to 8)
```

The following display appears:

```
Any Key to end...
Ax Good  Bad
#  Cycles     Cycles
N  GGGGGGGGG  BBBBBBBBB
```

If a single axis is selected, then the number of good and bad communication cycles will be constantly updated. If all cards are selected, then the display will remain blank until a key is pressed to terminate the test. At that time, the results of all axes will be displayed.

```
Any Key to end. .
Ax Good  Bad
#  Cycles     Cycles
1  GGGGGGGGG  BBBBBBBBB
2  GGGGGGGGG  BBBBBBBBB
3  GGGGGGGGG  BBBBBBBBB
4  GGGGGGGGG  BBBBBBBBB
5  GGGGGGGGG  BBBBBBBBB
6  GGGGGGGGG  BBBBBBBBB
7  GGGGGGGGG  BBBBBBBBB
8  GGGGGGGGG  BBBBBBBBB
```

The cycle test counters can count up to 2 billion test cycles, which typically would be about 200 days of continuous testing.

If a communication problem exists with the axis card, ie. all of the communication control lines do not indicate a healthy communication, then the following will appear:

```
********* Bad Communication *********
```

SELECTION E - Teach Pendant Input/Output Test

When this option is selected, the screen displays:

TEACH PENDANT TEST

This test procedure leads the user through several different inputs and outputs that are part of teach pendant function. Each pendant switch is tested by the user with the result being displayed. The potentiometer feedback value is displayed, and the READY lamp is flashed to indicate correct operation. In addition, the AUTO-START switch is tested. Each test is headed by a message prompting the user for the correct action.

a) READY LAMP!! - "E" to end

The READY lamp will flash until the user enters the E character, or until the test times out, which is approximately 1 minute.

b) GRIPPER OUTPUT!! - "E" to end

During this section of the diagnostics, the pneumatic gripper solenoid will be tested. On A150/A250 robots, the gripper solenoid can be heard easily, verifying correct operation. On other systems, the voltage must be measured at the proper test point to verify correct operation.

Input states - "E" to end

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b

The direction switch inputs are all scanned. All bits ("b" above) will appear as 1 or 0 as a result. Normally, the inputs will show a 1, or high value. The assignment of these inputs are described in the technical manual, and in the service manual.

c) Press AUTOSTART

Although not a teach pendant function, the diagnostic test waits until the front panel auto start switch has been toggled.

d) Press TEACH

The system waits until the TEACH push button has been pressed and released.

e) Press ABORT

The system waits until the ABORT push button has been pressed and released.

When selection 'A' is made, the system prompts the user to enter the current time and date information. Time is entered in 24 hour military format. The 'DAY' code is entered as a number from 1 to 7 representing Monday through Sunday. Once entered, the chip is set and is removed from its internal low power consumption mode where the internal oscillator is not active. The timer circuit is activated from this point on.

Selection 'B' de-activates the timer circuit and the clock stops. This useful only before removing the chip and placing it into its internal low-consumption mode for storage.

Selection 'C' displays the timer contents repetitively. If the seconds register fails to change with time, then repeat step 'A'. Continuing failure of this test indicates a failed timer chip, or the chip does not reside in the proper socket.

Selection 'D' returns to the main diagnostic level.

SELECTION M - Monitor Commands

The monitor is a low level control language which can be used to check, change and run the contents of the controller memory. Five one-letter monitor commands are available in this mode. The menu which appears when 'M' is selected lists these commands:

```
"S"UBSTITUTE
"D"UMP
"I"NPUT
"O"UTPUT
"G"O
"E"XIT
MON>>
```

The "MON>>" prompt will re-appear after each command is executed. When "E" is pressed, the main menu will re-appear.

The Substitute and Dump commands use the data types shown in Table A1.

The address of the data to be examined or modified must be entered in standard Intel Format of segment:offset.

In the Substitute command, once the data has been entered, a <CR> will terminate the command, but a <SPACE> will prompt the next data location to be changed. Never enter a blank line of data, as it will be read as "0".

The Dump command prompts for the number of data elements to be displayed. Both the DUMP and the SUBSTITUTE commands display and expect data in decimal format.

The INPUT and OUTPUT commands read and write hexadecimal data values to and from the 8086 physical I/O space. Note that unlike the DUMP or SUBSTITUTE commands, hexadecimal values are displayed.

Assuming that the diagnostic mode has been entered successfully, a variety of options are available. The options are selected according to the letter provided. The following list of options is displayed:

```
ENTER ONE OF THE SELECTIONS BELOW:
A = PIC R/W TEST
B = PIT COUNTING TEST
C = PIT INTERRUPT CAPABILITY TEST
D = 8087 MATH CO-PROCESSOR TEST
E = TEACH PENDANT I/O TEST
F = ANALOG INPUT TEST
G = ANALOG OUTPUT TEST
H = DIGITAL INPUT TEST
I = DIGITAL OUTPUT TEST
J = AXIS CARD TEST
K = MEMORY TEST
L = RTC TEST
M = MONITOR
N = EEPROM TEST
O = ALL TESTS
P = EXIT DIAGNOSTICS
SELECTION?
```

SELECTION A - Programmable Interrupt Controller Test

When this option is selected, the screen displays:

```
PIC R/W TEST
```

This test provides data bus and chip addressability requirements. Status words are read and written to the 8259 chip. Results of the test will show PASS or FAIL. Any read/write failure will be shown as:

```
PIC R/W FAIL WRITTEN ww:
READ: rr
```

Where ww and rr are byte values written to and read from the device.

SELECTION B - Programmable Interval Timer Test

When this option is selected, the screen displays:

```
PIT COUNTING TEST
```

Similar to test A, this test provides data bus and chip addressability requirements. Status words are read and written to the 8254 chip. Timer values are written into the chip, and elapsed counter values are compared to nominal

TOKEN: /116

FORMAT:

@DIGIO <MODE>

DESCRIPTION:

This command enables the system programmer to select the number of physical inputs and outputs that are available for system integration. This function is useful only when the COMBO/32 option is installed, since the basic 16 inputs and outputs are not programmable.

The MODE argument is a number from 1 to 8 inclusive that determines the operating characteristic of all 6 banks of programmable I/O. To indicate the I/O bank number and its mode of operation, the table uses the letter 'O' for Output and 'I' for Input followed by a number. The number indicates the bank number. The standard I/O uses bank 0 and 1 (there are 16 standard inputs and outputs). Each bank has 8 inputs and 8 outputs. Consequently, the programmable I/O on the combo card starts at bank 2. The default condition of the combo card I/O is mode #0. The table below indicates the operating mode of all of the combo-card banks with any given mode input:

This translates to which cable connection at the controller will provide the proper signals. Since each connector handles 16 I/O points, there are three groups of 16 I/O. The letters A, B and C correspond to the connectors on the back panel labelled 17-24, 25-32, 33-40 respectively. These connector designations will no longer make sense if any other mode but 1 or 5 is programmed.

Mode	Port ID					
	A0	B0	C0	A1	B1	C1
1	O2	O3	O4	I2	I3	I4
2	I9	I8	I7	I2	I3	I4
3	O2	I8	I7	I2	I3	I4
4	O2	O3	I7	I2	I3	I4
5	O2	O3	O4	I2	I3	I4
6	O2	O3	O4	I2	I3	O5
7	O2	O3	O4	I2	O6	O5
8	O2	O3	O4	O7	O6	O5

TABLE A-3 Default I/O modes after Teach Start

TOKEN: /

FORMAT:

@@DIAG

DESCRIPTION:

This command permits the system programmer to interrogate the operation of the controller. This should be done when the controller is considered to have some form of failure.

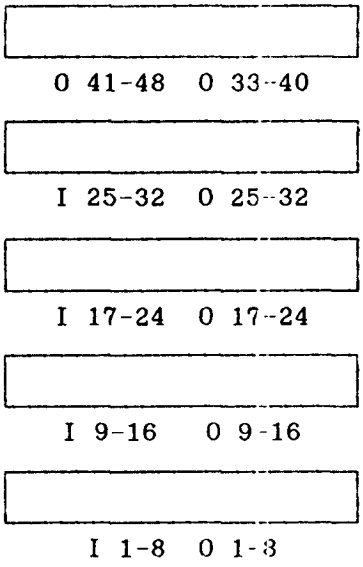
WARNINGS:

1. After diagnostics have been run, the controller should be turned OFF, then back ON again. This will clear any pending diagnostic operations and will return the controller to the state of readiness.

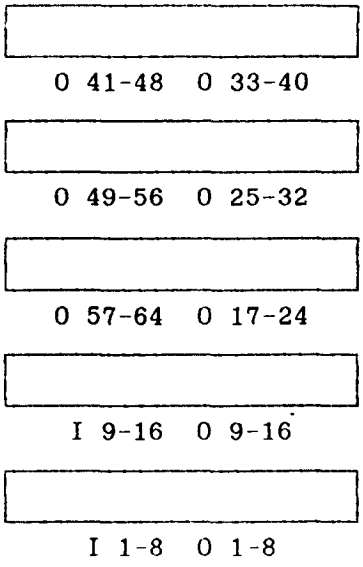
2. The diagnostic procedure will request that the arm power be turned off before the diagnostic procedures begin. **FAILURE TO TURN THE ARM POWER OFF MAY CAUSE UNEXPECTED ARM MOTIONS AND POTENTIAL DAMAGE TO PEOPLE AND MACHINES COULD RESULT.**

3. The diagnostic test procedure should be exercised by, or under the guidance of, trained service personnel.

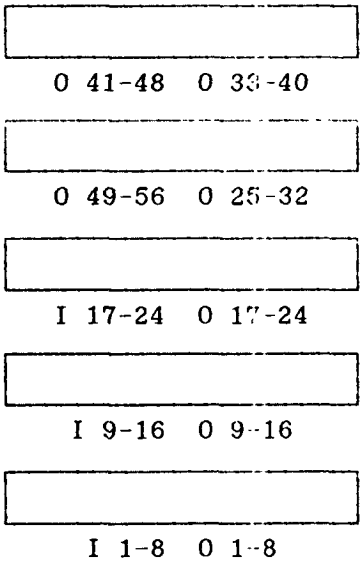
MODE 6



MODE 8



MODE 7



APPLICABLE MODES:
{I},{M},{P}

TOKEN: /019

FORMAT:

@@CAL

DESCRIPTION:

Factory calibrate command used at Factory. Do **NOT** use this command unless servicing the robot.

WARNING

DO NOT USE THIS COMMAND, UNLESS ON THE ADVICE OF CRS PLUS.

EXAMPLE

Step	Action
1	Do a "TEACH" start.
2	Arm power should be on.
3	Robot should be in "MANUAL" mode.
4	Move arm to zero position as shown in figure A-1.
5	Enter "NOMANUAL".
6	Enter "PASSWORD".
7	Support arm so it doesn't drop during zeroing.
8	Enter "@ZERO".
9	Enter "JOINT 2,90"
10	Check each joint to ensure they are in "HOME" range. If not repeat.
11	Enter "@@CAL Y".

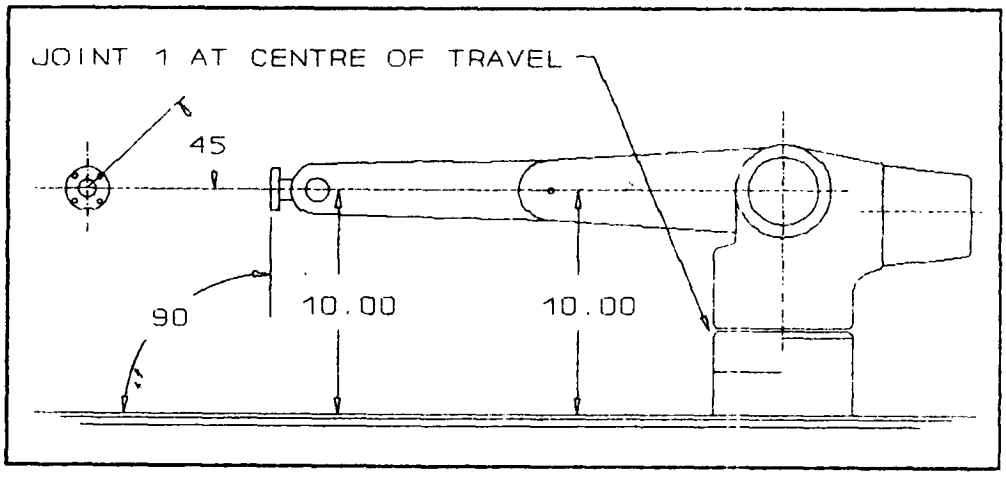


Figure A-1 Pose of the arm at true "zero" (CALRDY Pose)

TOKEN: /180

FORMAT:

@GANTRY

NOTE:

THIS COMMAND IS USED WITH A150/A250 ROBOTS ONLY. IF THIS COMMAND IS USED WITH ANY OTHER ROBOT CONFIGURATION, AN ERROR WILL BE ISSUED.

DESCRIPTION:

This command sets up a GANTRY condition. This means that every cartesian location includes the position of axes 6 and 7 as X and Y gantry coordinates.

The coordinates of the gantry are stored as a real numbers in 8 bytes immediately above the standard six coordinates of each cartesian entry in the location table. The real values are the "joint" engineering unit values of the axes when the HERE command is issued. Use of @XRATIO, @XLIMITS, and @XPULSES for both axes 6 and 7 is mandatory before issuing this command. Before storing values or commanding motion after the GANTRY command is issued, an XHOME command must have been issued for both axes.

The coordinates stored for the extra axes are entered only as an input to the motion command. Shifting of these coordinates is possible using the SHIFTA command. Entry of a point "off-line" using the POINT command will permit entry of the axis values. All motion commands referring to cartesian locations after issuing the GANTRY command will coordinate with the GANTRY axes.

For long moves involving the gantry axes it may be an advantage to use the SLEW mode. This is entered with the ENABLE SLEW command.

GANTRY is reset using the "TRACK RESET" command.

APPLICABLE MODES:

{I},{M},{P}

CROSS-REFERENCES:

RAPL-II COMMANDS: TRACK, @XPULSES, @XRATIO, @XLIMITS, SHIFTA

OTHER CRS Plus Publications: Technical Manual APPENDIX J

TOKEN: /019

FORMAT:

@@CAL

DESCRIPTION:

Factory calibrate command used at Factory. Do **NOT** use this command unless servicing the robot.

WARNING

DO NOT USE THIS COMMAND, UNLESS ON THE ADVICE OF CRS PLUS.

EXAMPLE

Step	Action
1	Do a "TEACH" start.
2	Arm power should be on.
3	Robot should be in "MANUAL" mode.
4	Move arm to zero position as shown in figure A-1.
5	Enter "NOMANUAL".
6	Enter "PASSWORD".
7	Support arm so it doesn't drop during zeroing.
8	Enter "@ZERO".
9	Enter "JOINT 2,90"
10	Check each joint to ensure they are in "HOME' range. If not repeat.
11	Enter "@@CAL Y".

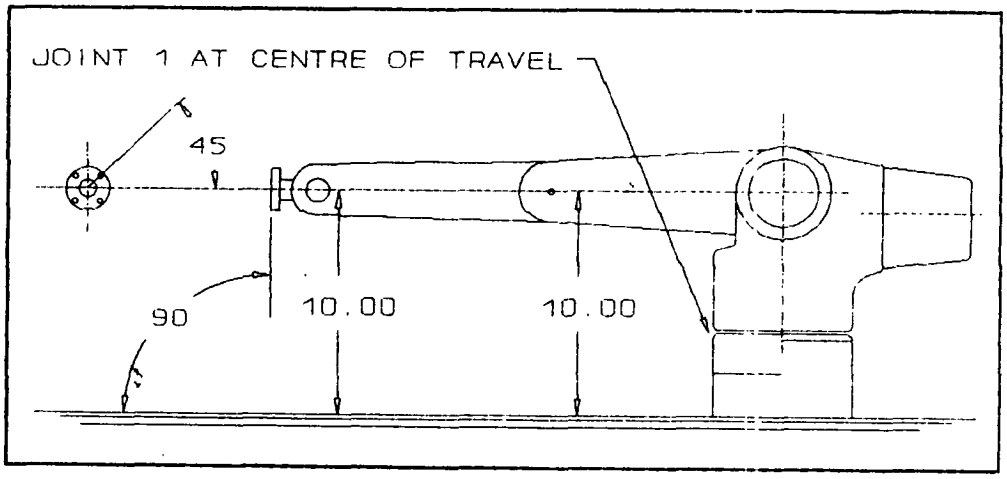


Figure A-1 Pose of the arm at true "zero" (CALRDY Pose)

TOKEN: /178

FORMAT:

@INIT <AXES|AXIS|DARTA|DARTB|PIC|PIT|VARS|LOCS|PROGS|ALL>

DESCRIPTION:

This command initializes a part of the controller hardware. This should only be done in the event of a failure, and under direction of a qualified service technician.

THIS COMMAND IS NOT FOR GENERAL USE.

AXES or **AXIS** selection will initialize all or one selected smart axis card. **VARS**, **LOCS**, **PROGS** will initialize the appropriate table. **ALL** will initialize all three tables. **DARTA**, **DARTB**, PIC, **PIT** initialize specific hardware components on the mother board.

APPLICABLE MODES:

{I},{M},{P}

CROSS-REFERENCES:

@ACCEL

TOKEN: /013

FORMAT:
@ACCEL (axis#),<value>

DESCRIPTION:

Alter the maximum acceleration rate parameter for any axis. This value can be (0 < value <= 2) and is a real number.

The unit of <value> is in pulses per tick/tick, where a tick is the minimum unit of time used by the command generation software and a pulses is the basic length unit of the servo.

For the A150 robot with standard axis cards, the default value is 1 ie. 1/1000 of a motor turn per 0.004^2 seconds^2. This gives a maximum acceleration of 62.5 revolutions/sec^2.

For the A250 controller the time base for this calculation is much shorter. The default value is 0.0493 1/1000ths of a motor rev. per $.001^2$ seconds^2. This gives an acceleration of 49.3 revolutions/sec^2. The system should not exceed 120 revolutions/sec^2 which is .12 units.

APPLICABLE MODES:

{I},{M}

TOKEN: /162

FORMAT:

@LOFB <SET[<Limit>,<Time_out>]|RESET>

DESCRIPTION:

This command sets or resets the Loss Of FeedBack (collision or servo failure) detection check. When set, this function will issue an error for any axis which is being commanded at the programmed limit for more than the programmed time with no change in position. These symptoms are typical of the following situations:

1. Broken encoder, producing no feedback.
2. Blown fuse.
3. Robot has collided with an object and is binding.
4. Static overload.

The values of limits and time-out can be set in A250 controllers only (ie. not in A150 controllers). The units of <Limit> are in pulses while the units of <Time_out> is in milli-seconds.

DEFAULT CONDITION

For version 3.4.5 and 3.4.6, the default condition is off (Reset). For all versions from 3.5.0 and on, and RAPL-II, the default condition is on (Set). The default limit and time values in RAPL-II are 115 pulses and 400 msec.

APPLICABLE MODES:

{I},{M}

EXAMPLE:

1. To activate collision detection

```
>>@LOFB SET<CR>
>>
```

2. To de-activate collision detection

```
>>@LOFB RESET<CR>
>>
```

3. To check current limits and set a higher sensitivity limit

```
>>@LOFB<CR>
SET
 POSITION ERROR IN PULSES:   +115
 TIME DURATION (MSEC): 00500

>>@LOFB SET,75,400<CR>
>>
```

CONTENTS

TOKEN: /167

FORMAT:

@MAXVEL [axis#],[VELOCITY]

DESCRIPTION:

This command is used to tune the maximum value of command velocity to the maximum pulse output rate of the encoder for an extra axis.

If this value is set improperly, the encoder of the extra axis may not be anble to send out pulses fast enough to keep up with the command. The axis may lag behind so badly that the command error may overflow its buffer and the motor will reverse. In less drastic cases, the motor will be poorly synchronized with the robot, particularly at higher command speeds, and it will be difficult to control the axis under manual control.

Determine the value of [VELOCITY] by running the motor at full speed under the given load and amplifier conditions, and measure the encoder speed in RPM. The encoder output can be used as a tach if a scope can be attached to its output. The frequeny along with the opulse count per revolution will give the speed in RPM using the following formula:

$$\text{ENCODER RPM} = \frac{60}{(\text{Pulses/rev})(\text{Pulse-Period}_{sec})}$$

WARNING:

The @XMAXVEL command must be executed after any @XPULSES or @XRATIO command, since the @XMAXVEL command generates internal parameters that are dependant upon all extra axis parameters.

APPLICABLE MODES:

{I},{M}

9-4 AUTO START PROGRAM (Continued)

The E_STOP subroutine is called whenever a safe condition is breached in the workspace, or when the normal end of the job occurs. This routine first moves the robot to a safe location (SAFE), The IGNORE command causes the robot to ignore the stop signal which caused it to jump to this routine. It then moves back into the homing bracket, and the ARM POWER is turned off.

```
PROGRAM E_STOP
10 MOVE SAFE
20 IGNORE
30 MOVE #ATCAL
40 MOVE #PREHOME
50 SPEED 10
60 MOVE #FIXTURE
70 FINISH
75 DELAY 1
80 DISABLE ARM
90 STOP
$
```

In this program, the location #ATCAL is the position of the arm where the homing sequence is called. Using a precision point in all these locations is recommended here as changing the TOOL transform (if say the tool length changes slightly) will re-position the arm and may cause a collision with the bracket.

#PREHOME is a location just above the bracket itself, and #FIXTURE is the position of the robot in the homing bracket.

Notice that turning the arm off is permitted in a program, however an error will result, stopping the program. This has no effect on the operation of this program.

TOKEN: /122

FORMAT:

@NOA

DESCRIPTION:

This command displays the current number of axis, and permits the user to change the number.

DEFAULT CONDITION

At power up, the controller reads the number of axis cards installed. If there are more or less than expected the system will prompt the user to enter the correct number. The system will not permit further operation until a response has been entered. If this sequence is aborted, the RAPL-II interpreter will be entered, but arm power will not be enabled.

APPLICABLE MODES:

{I},{M}

having to measure the true zero location as described in the Technical Manual, page A-5.

The robot must then be moved into the home bracket. This is most easily done with the ARM POWER off. When the arm is in position, turn arm power back on. set the speed to a slow value like 10%. Issue a joint command to move the robot out of the bracket such as "JOINT 3,20". Watch the arm carefully to make sure it leaves the bracket easily without binding. Application of a bit of grease to the inner edge of the bracket can ease the robot's exit.

Once the robot has exited the bracket, it must have space to perform the Homing sequence during which it moves slightly. The position arrived at after the 20 degree joint 3 move may not be acceptable due to interference with external equipment etc. In this case, use further JOINT commands to move the arm to a clear location. Record the sequence of moves taken to get clear as the robot will have to repeat them each time it is homed. Once at a clear location, use the @@CAL command to recalibrate the robot at the new home location.

There is one note of caution to be added. If during the calibration sequence, an individual move of a joint to get to the encoder index mark takes very little time to complete, that joint should be commanded to move back by about 1/2 a motor turn (200 to 500 pulses) to ensure that it sees the correct index mark every time.

In addition, if the joint takes almost a complete turn of the motor to complete, the joint should be compensaated forward by about 200 to 500 pulses. Such a correction should be used in the homing program (see below).

TOKEN: /134

FORMAT:

@@RE

DESCRIPTION:

This command enables the ACI software. This will dedicate the serial port #1 to the ACI software. No other normal serial input will be allowed on this channel. Output is still allowed, but it will cause problems if an ACI cycle is concurrently running.

APPLICABLE MODES:

{I},{M}

CHAPTER 9 - HOMING BRACKET INSTALLATION

9-1 INTRODUCTION

The purpose of the Homing Bracket is to allow simple homing of the robot in a situation where the operator may not be trained for (or be able to) home it accurately using the factory-supplied marks. Such applications are found in industrial systems and labs where the arm may not be easily accessible.

In order to use the homing bracket, the robot calibration must be changed. This is not difficult and will be described in detail in this section of the manual.

9-2 HOMING BRACKET INSTALLATION

The position of the bracket in the robot workspace is important to the success of the procedure. The robot must exit from the bracket using joint or motor commands only. This is because until it is homed, the robot can only make moves which do not depend on the world coordinate system. The most convenient joint to use for exiting is joint 3. The wrist flange must be perpendicular to the upper arm in order to slide cleanly out of the bracket. The angle of the support plate on the bracket is designed to position the bracket as close to the robot as practical and still allow room for typical tooling. The optimum position can be seen in Figure 9-1.

In addition to the bracket itself, the small fixture plate must be installed on the robot wrist flange. Any existing tooling must be removed and the plate installed between the gripper flange and the tooling. It should be installed so that the narrow end would be at the bottom when the robot is at the READY position.

Adding this plate between the robot and the gripper changes the position of the arm in accessing locations in the robot task. The distance can be compensated by re-teaching the locations or by adding a tool offset of 0.187 in the X direction. If no tool offset is used already, enter a location NEWTOOL with the value ".187,0,0,0,0,0". Then issue the command "TOOL NEWTOOL". Re-teaching the locations is usually a good idea any time the gripper comes off the robot, as it may not be repositioned exactly where it was before.

TOKEN: /135

FORMAT:

@@RH

DESCRIPTION:

This command displays the last ACI header which was read in. This will indicate the nature of the last (or current) ACI transaction.

APPLICABLE MODES:

{I},{M}

```
char Name[8];
long AxisVal[8];
int  Checksum;
 } *LocationPointer;

CartesianLocation struct
 {
char Name[8];
float Coord[8];
int  Checksum;
 }  *LocationPointer;
 char ProgramBuffer[ProgramBufferSize];
    char *ProgramBufferPointer;
```

The checksums associated with each element of the structures consists of the arithmetic sum of each byte in the structure element. Only the low byte of the checksum is used. The checksum used in the program table is the checksum of the entire program that it references, from the first byte of the program, up to and including the '$' delimiter used to end a program.

Exact addresses for each of the system parameters mentioned above can be found in the memory map for the current version of RAPL-II. This information can be found in Appendix G.

TOKEN: /137

FORMAT:

@@RN

DESCRIPTION:

This command selects a slave device number for the robot controller. This value can be any value between 1 and 127. The value of 0 is not permitted.

APPLICABLE MODES:

{I},{P},{M}

The Program Buffer

The program buffer is a contiguous byte array which contains the user programs. The length of this buffer is set when the ALLOCATE command is entered. Its size depends on what is left of the user memory after allowing for Program, Location and Variable tables, as well as the Reserved memory.

The program buffer stores programs as sequential ASCII characters. Program information is entered by typing in through the user terminal, or downloading from a Master computer using the ACI interface and support software, such as the ROBCOMM software system.

The location of the start of a program is determined by RAPL-II from the index in the program table, relative to the program buffer pointer. Each program is terminated by a dollar-sign character ($) which acts as a seperator. Memory in the program buffer is altered dynamically whenever editing of a program takes place. The memory is "squeezed" up or down as lines are entered, modified, or deleted. Thus the index of all entries in the program table will have to change whenever a line is entered into a program located below the one changed. RAPL-II makes these changes unseen by the programmer. The program buffer can be subdivided into two sections immediately after an ALLOCATE command. The HIMEM command will partition an area away from the uppermost program buffer. This reserved memory will then remain untouched for special programming applications, or PCP's.

Variable Storage

Variables are located in the variable table. The length of this table is defined when the ALLOCATE command is entered. Like the program table, there are three fields for each variable entry in the table The first is the eight character string used to store the name. The second is a four byte real value which stores the value of the variable. The third is a word quantity used to store the checksum.

Location Storage

Locations (both precision, and cartesian types) are stored in a location table containing ten fields for each location. The first is the 8-character name string. The next eight fields are four-bytes each used to store the components of each location. The RAPL-II controller can control up to 8 individual axes of motion. A precision point can use position information for up to eight axes. In this case, each of the eight fields store a double size integer value of each axis motor coordinate (in encoder pulse units). When a cartesian coordinate is stored, it uses the first six of these 8 fields to store a four byte real number representation of each world component. If a GANTRY command has been specified, the

TOKEN: /114

FORMAT:

@SAVE

DESCRIPTION:

This commands permits the programmer to save RAPL-II user memory to the EEPROM bank, which is available when the option is installed. Before the @SAVE command is used, the write protect switch on the top of the EEPROM card must be selected to the write enable position. This is done by removing the controller lid, and sliding the switch with a small screwdriver. Once the @SAVE function is completed, slide the switch back to the original write protect setting.

WARNING:

Data loss will result if the write protect switch is not returned to the write protect setting after the @SAVE is completed and before the next time the controller is turned ON.

APPLICABLE MODES:

{I},{M},{P}

EXAMPLES:

1. Save the complete user memory
   >> @SAVE

CROSS-REFERENCES:

@RESTORE, ALLOC, FREE

4) VIA path.

The VIA command defines a curve through space based upon a set of intermediate points that will be approximated, but the robot may not move through them, depending upon the acceleration constraints of the robot. The CARTVEL flag determines the mode of velocity control, as before. When CARTVEL is enabled, the path between points is generated as a straight line. With the A150 series, the VIA command is allowed only when CARTVEL is disabled. IA path commands are generated at 16 millisecond intervals for joint interpolated speed profiling, or 32 milliseconds with cartesian velocity profiling selected with the CARTVEL command.

Command generation can be halted using the RAPL-II "HALT ON <[-]Input#>" command and enabling the HOLD parameter. When an interrupt occurs, all joints deccelerate at the rate determined by the characteristics of the filter. When the interrupting signal returns to normal, all joints resume motion also at the rate dictated by the filter stage. This input should be treated as an emergency stop feature, and should not be used as a normal motion hold.

A150 Controller

All command generation output is digitally filtered to provide an update rate of 0.0035 seconds per update for each servo axis.

A250 Controller

Joint interpolated motion is handled entirely by the joint controllers. All other modes of path generation are provided by the motherboard 8086/8087 tandem processors. High levels of motion smoothness is provided by additional compensation by the axis cards, before the commands are sent to the motors. This compensation provides higher performance in terms of speed and smoothness.

TOKEN: /180

FORMAT:

@TRACK <X|Y|RESET>

NOTE:
THIS COMMAND IS USED WITH A150/A250 ROBOTS ONLY. IF THIS COMMAND IS USED WITH ANY OTHER ROBOT CONFIGURATION, AN ERROR WILL BE ISSUED.

DESCRIPTION:

This command sets up a TRACK condition. This means that every cartesian location includes the position of axis 6 - the track position.

The coordinate of the track is stored as a real number in 8 bytes immediately above the standard six coordinates of each cartesian entry in the location table. The real value is the "joint" engineering unit value of the axis when the HERE command is issued. Use of @XRATIO, @XLIMITS, and @XPULSES are mandatory before issuing this command. Before storing values or commanding motion after the TRACK command is issued, an XHOME command must have been issued.

The coordinate stored for the extra axis is entered only as an input to the motion command. Shifting of this coordinate is possible using the SHIFTA command. Entry of a point "off-line" using the POINT command will permit entry of the track axis value. All motion commands referring to cartesian locations after issuing the TRACK command will coordinate with the TRACK axis.

For long moves involving the gantry axes it may be an advantage to use the SLEW mode. This is entered with the ENABLE SLEW command.

The TRACK condition is turned off using "TRACK RESET" command.

APPLICABLE MODES:

{I},{M},{P}

WARNINGS:

1. Accessing any locations taught without the TRACK option may cause unexpected results.

CROSS-REFERENCES:

RAPL-II COMMANDS: GANTRY, @XPULSES, @XRATIO, @XLIMITS, POINT, SHIFTA

OTHER CRS Plus Publications: Technical Manual APPENDIX J

8-4 PATH CONTROL - COMMAND GENERATION

The A150/A250 series of controllers have multiple path generation modes which can be summarized as three major modes of path control:

1) Joint Interpolated motion

MOVE, APPROACH, DEPART, MOTOR, JOINT, MI, MA are RAPL-II commands that create joint interpolated motion. In the A150 controller, the joint interpolated command rate is 4.5 milliseconds. In the A250 controller, the joint interpolated motions are performed at 1 millisecond intervals within the IAXIS axis cards. The Joint Interpolated motion guarantees that all joints to be moved will start and stop together. It also guarantees that maximum acceleration and decceleration will not be exceeded for each axis. Command updates occur at four milli-second intervals. Figure 8-2 shows typical acceleration, velocity and position profiles for a joint interpolated robot motion.

2) Straight line Motion

JOG, straight line MOVE, DEPART, APPROACH, JOG commands provide a straight line path in cartesian space. Cartesian velocity can be controlled by issuing the ENABLE CARTVEL command, which will process the speed command as a percentage of the maximum cartesian velocity (@MAXSPD command). Otherwise, speed control is based on maximum joint velocity. In Straight Line, command updates are performed at a slower rate (18 milliseconds for A150 controller, 16 milliseconds for the A250 controller). This mode of control is a specialized form of the PATH control described below. The "knots" are calculated by RAPL-II based on a linear interpolation between end points. The number of knots used for the straight line is normally 10, which gives reasonable accuracy to the path. If a higher processing speed is required, the number of knots may be reduced. The ROBCOMM package with its memory edit feature can be used for this. Acceleration and decceleration in this mode are based on world coordinates instead of individual joints. If wrist motions are required during the straight line path, then the wrist rotational speeds are continuous throughout the straight line path segment. Joint limit checking is performed before the straight line move, and if any joint exceeds the programmed limits, the command is immediately terminated, and an error message is displayed.

3) Continuous path

CPATH, CTPATH, GOPATH. Defining a path curve through space by selecting a number of intermediate points through which the robot will move. Again, the selection of the CARTVEL parameter will determine the mode of velocity control, but the algorithm always generates a path through the intermediate points.

TOKEN: /168

FORMAT:

@XLINKS <Axis #>,<Length>

DESCRIPTION:

The link lengths of the standard transformations can be altered by this command. This is useful when production robots must be exactly calibrated in order to achieve accurate cartesian transformations. Axis number and link length data can be entered as constants or variable references.

APPLICABLE MODES:
{I},{M},{P}

CROSS-REFERENCES:
@XRATIO, @XPULSES

B-4 HEADER SEQUENCE

The header block consists of a description of the data transfer that will take place. The header block is broken down into the following byte description:

1	SOH	Start Of Header character.
2	SLAVE ID + 20h	Slave identification number.
3	MASTER ID + 20h	Master identification number.
4	READ/WRITE	Data read or write indicator.
5	MEMORY TYPE	Memory access type (see special codes).
6	NUMBER OF FULL BLOCKS	Number of full blocks (128-bytes) to be transferred.
7	NUM. BYTES IN LAST BLOCK	Number of bytes in last block.
8	MEMORY OFFSET LOW BYTE	Target memory starting address
9	MEMORY OFFSET HIGH BYTE	"
10	MEMORY SEGMENT LOW BYTE	"
11	MEMORY SEGMENT HIGH BYTE	"
12	ETX	End of Text character.
13	LRC	Longitudinal redundancy check.

Table B-2 - Header Contents

All data values are expressed in hexadecimal notation unless otherwise noted.

The slave ID number is the same as appears in the enquiry sequence. The master ID must be '01' as this is a single-master system.

The READ/WRITE byte identifies the direction of memory transfer. A WRITE operation will transfer memory from the slave (robot) to the master (host). In effect the request is "WRITE to master requested". The READ command is the opposite. It transfers memory from the master to the slave. The following codes are used:

READ	**01h**
WRITE	**00h**

The memory type specifier identifies specific areas of memory to/from which the data transfer is to take place for special transfers. In such transfers, the burden of providing specific memory addresses is removed from the programmer. A list of all special codes is contained in section B-12.

The code '00h' is the general purpose memory type code. This code enables the programmer to read or write to any byte in the 8086 memory space. This makes the ACI a very powerful (and potentially dangerous) tool to work with.

TOKEN: /161

FORMAT:

@XRATIO <Axis#>, <Factor>[,<R>]

DESCRIPTION:

This function provides a scaling factor used to convert motor pulses to units of measurement. For the A150/A250 robot, this command is allowed on axis 6, 7, or 8 only, as the scaling of the 5 robot axes must be fixed. For motion controllers, all 8 axes can be configured.

The XRATIO value, along with the XPULSES value when multiplied together form the complete conversion from motor pulses to units of measure. The units of XRATIO is then in terms of encoder revolution per engineering unit.

The XRATIO command contains a final optional argument that will tell the system whether or not the joint is a revolute joint, or a linear joint. If the final ',R' argument sequence is entered, then the system knows that the engineering units of the XRATIO command are RADIANS. All joint angle limits must be entered as radians as well. However, the joint angle displays, and the joint angle values in the JOINT command will be interpreted in DEGREES, and not radians, for convenience. In addition, the joint position displayes will be shown in degrees as well.

DEFAULT CONDITION

For A150/A250 Robots:

The default value of TR for axes 6 through 8 is 1.

APPLICABLE MODES:

{I},{P},{M}

B-1 INTRODUCTION

The ACI permits external computer systems to communicate with one or more robot systems on a single serial (RS232 or RS422) link.

This protocol is used to transfer data either to or from the master device. It allows any of the addressable 8086 memory to be the object of the communication using the 8086 standard segmented addressing technique. It performs error checking and automatic transmission retries in order to ensure a reliable communication link.

The ACI is a master/slave protocol. All robot controllers are configured as slaves in the network. Any external; computer would then be configured as the master. Only a master can initiate communication.

B-2 MASTER PROTOCOL

A communication consists of four separate blocks. they are described in detail in the following sections.

1) The master device establishes a communication link by specifying a slave device number as a target. The slave confirms its existence on the serial channel, and the master confirms to the target that it is indeed requesting a communication.

2) After the link has been established, the master provides the slave with the information describing the data transfer that is about to take place.

3) After the reception of the data transfer specifications is confirmed by the slave, the actual data is sent. The data is transferred in packages (blocks) of 128 bytes. Each block of data contains its own start/stop characters and a checksum test byte to validate the data after transfer.

4) After the data has been successfully received, the master closes the communication link with an EOT character.

TOKEN: /022

FORMAT:

@ZERO

DESCRIPTION:

This command sets the robot axis position registers to zero. This command is used when calibrating the robot.

WARNING:

Use this command with extreme care. Once the @ZERO command is issued, then the real travel limits of the arm may not be protected by the software limits.

TOKEN: /022

FORMAT:

@ZERO

DESCRIPTION:

This command sets the robot axis position registers to zero. This command is used when calibrating the robot.

WARNING:

Use this command with extreme care. Once the @ZERO command is issued, then the real travel limits of the arm may not be protected by the software limits.

B-1 INTRODUCTION

The ACI permits external computer systems to communicate with one or more robot systems on a single serial (RS232 or RS422) link.

This protocol is used to transfer data either to or from the master device. It allows any of the addressable 8086 memory to be the object of the communication using the 8086 standard segmented addressing technique. It performs error checking and automatic transmission retries in order to ensure a reliable communication link.

The ACI is a master/slave protocol. All robot controllers are configured as slaves in the network. Any external; computer would then be configured as the master. Only a master can initiate communication.

B-2 MASTER PROTOCOL

A communication consists of four separate blocks. they are described in detail in the following sections.

1) The master device establishes a communication link by specifying a slave device number as a target. The slave confirms its existence on the serial channel, and the master confirms to the target that it is indeed requesting a communication.

2) After the link has been established, the master provides the slave with the information describing the data transfer that is about to take place.

3) After the reception of the data transfer specifications is confirmed by the slave, the actual data is sent. The data is transferred in packages (blocks) of 128 bytes. Each block of data contains its own start/stop characters and a checksum test byte to validate the data after transfer.

4) After the data has been successfully received, the master closes the communication link with an EOT character.

TOKEN: /161

FORMAT:

@XRATIO <Axis#>, <Factor>[,<R>]

DESCRIPTION:

This function provides a scaling factor used to convert motor pulses to units of measurement. For the A150/A250 robot, this command is allowed on axis 6, 7, or 8 only, as the scaling of the 5 robot axes must be fixed. For motion controllers, all 8 axes can be configured.

The XRATIO value, along with the XPULSES value when multiplied together form the complete conversion from motor pulses to units of measure. The units of XRATIO is then in terms of encoder revolution per engineering unit.

The XRATIO command contains a final optional argument that will tell the system whether or not the joint is a revolute joint, or a linear joint. If the final ',R' argument sequence is entered, then the system knows that the engineering units of the XRATIO command are RADIANS. All joint angle limits must be entered as radians as well. However, the joint angle displays, and the joint angle values in the JOINT command will be interpreted in DEGREES, and not radians, for convenience. In addition, the joint position displayes will be shown in degrees as well.

DEFAULT CONDITION

For A150/A250 Robots:

The default value of TR for axes 6 through 8 is 1.

APPLICABLE MODES:

{I},{P},{M}

B-4 HEADER SEQUENCE

The header block consists of a description of the data transfer that will take place. The header block is broken down into the following byte description:

1	SOH	Start Of Header character.
2	SLAVE ID + 20h	Slave identification number.
3	MASTER ID + 20h	Master identification number.
4	READ/WRITE	Data read or write indicator.
5	MEMORY TYPE	Memory access type (see special codes).
6	NUMBER OF FULL BLOCKS	Number of full blocks (128-bytes) to be transferred.
7	NUM. BYTES IN LAST BLOCK	Number of bytes in last block.
8	MEMORY OFFSET LOW BYTE	Target memory starting address
9	MEMORY OFFSET HIGH BYTE	"
10	MEMORY SEGMENT LOW BYTE	"
11	MEMORY SEGMENT HIGH BYTE	"
12	ETX	End of Text character.
13	LRC	Longitudinal redundancy check.

Table B-2 - Header Contents

All data values are expressed in hexadecimal notation unless otherwise noted.

The slave ID number is the same as appears in the enquiry sequence. The master ID must be '01' as this is a single-master system.

The READ/WRITE byte identifies the direction of memory transfer. A WRITE operation will transfer memory from the slave (robot) to the master (host). In effect the request is "WRITE to master requested". The READ command is the opposite. It transfers memory from the master to the slave. The following codes are used:

READ 01h
WRITE 00h

The memory type specifier identifies specific areas of memory to/from which the data transfer is to take place for special transfers. In such transfers, the burden of providing specific memory addresses is removed from the programmer. A list of all special codes is contained in section B-12.

The code '00h' is the general purpose memory type code. This code enables the programmer to read or write to any byte in the 8086 memory space. This makes the ACI a very powerful (and potentially dangerous) tool to work with.

TOKEN: /168

FORMAT:

@XLINKS <Axis #>,<Length>

DESCRIPTION:

The link lengths of the standard transformations can be altered by this command. This is useful when production robots must be exactly calibrated in order to achieve accurate cartesian transformations. Axis number and link length data can be entered as constants or variable references.

APPLICABLE MODES:

{I},{M},{P}

CROSS-REFERENCES:

@XRATIO, @XPULSES

B-5 DATA BLOCK TRANSMISSION SEQUENCE

The data block transfer sequence is determined by the format specified in the header block.

In a data WRITE ('W') command, the slave will begin to write the specified data blocks quickly after it issues the **ACK** character in response to the header block. The slave expects an ACK character from the master after each block is transmitted. This depends on the successful check of the LRC value sent with each block as described below. If the LRC does not match the master will send a **NAK**, prompting a retry of the block transmission. After the last block is sent, the and acknowledged, the slave will issue an **EOT**. It then expects a final **EOT** from the master. When it receives that, it will close off communication. The data blocks are configured in the following manner:

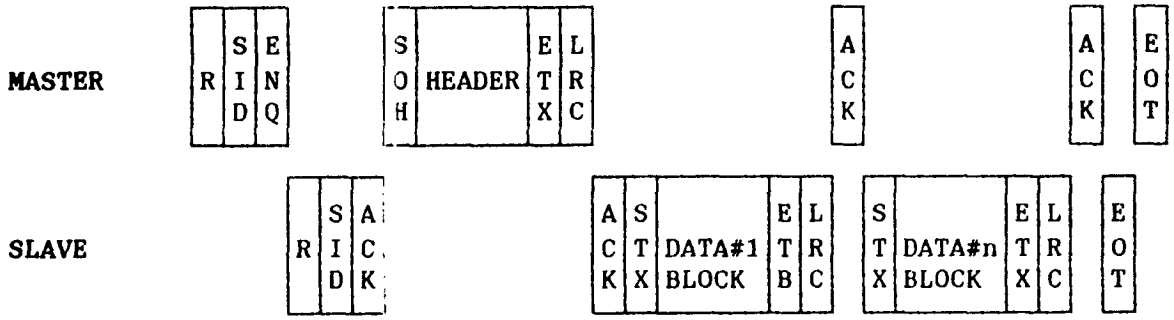
STX + [DATA BYTE #0] + ,....., + [DATA BYTE #127] + ETB + LRC

for the full data blocks and:

STX + [DATA BYTE #0] + ,....., + [DATA BYTE #n-1] + ETX + LRC

The last block uses an ETX character instead of an ETB character in order to signal the end of the transferred data.

The following diagram illustrates the flow of characters between the master and the slave in a data-write cycle:



TOKEN: /180

FORMAT:

@TRACK <X|Y|RESET>

NOTE:
THIS COMMAND IS USED WITH A150/A250 ROBOTS ONLY. IF THIS COMMAND IS USED WITH ANY OTHER ROBOT CONFIGURATION, AN ERROR WILL BE ISSUED.

DESCRIPTION:

This command sets up a TRACK condition. This means that every cartesian location includes the position of axis 6 - the track position.

The coordinate of the track is stored as a real number in 8 bytes immediately above the standard six coordinates of each cartesian entry in the location table. The real value is the "joint" engineering unit value of the axis when the HERE command is issued. Use of @XRATIO, @XLIMITS, and @XPULSES are mandatory before issuing this command. Before storing values or commanding motion after the TRACK command is issued, an XHOME command must have been issued.

The coordinate stored for the extra axis is entered only as an input to the motion command. Shifting of this coordinate is possible using the SHIFTA command. Entry of a point "off-line" using the POINT command will permit entry of the track axis value. All motion commands referring to cartesian locations after issuing the TRACK command will coordinate with the TRACK axis.

For long moves involving the gantry axes it may be an advantage to use the SLEW mode. This is entered with the ENABLE SLEW command.

The TRACK condition is turned off using "TRACK RESET" command.

APPLICABLE MODES:

{I},{M},{P}

WARNINGS:

1. Accessing any locations taught without the TRACK option may cause unexpected results.

CROSS-REFERENCES:

RAPL-II COMMANDS: GANTRY, @XPULSES, @XRATIO, @XLIMITS, POINT, SHIFTA

OTHER CRS Plus Publications: Technical Manual APPENDIX J

B-7 ERROR CHECKING AND RECOVERY

The ACI can handle a range of different communication errors. The two broad classifications of errors are transmission inaccuracies and character time-outs.

When a character is transmitted incorrectly, two things may happen. First, if it results in a checksum error, the data block is retried. This process continues for a programmed number of retries. If a correct transmission cannot be established, the slave executes a complete cycle reset, and is ready for the next communication attempt. On the other hand, if a control character is mis-read because of a transmission error, the cycle is aborted immediately, and the ACI is reset.

Character time-outs are used when a pending communication cycle has been interrupted by an overly long time delay. After a time-out condition, the slave ACI issues an EOT then resets for another cycle. A time-out error causes an immediate reset of the ACI.

The following list shows the errors possible at the slave end of the ACI link. The error can be seen using the password-protected @@RS (Remote Status) command.

Error Number	Error Description, Cause, and Recovery
1	Not Used
2	No SOH character found to lead off header sequence. The SOH character must be the first character of the header. Any other character will cause the ACI to terminate and the communication cycle will be aborted. The slave will issue an EOT character.
3	The ACI encountered a software error when reading in the header block. This is an internal software failure. Contact your CRS representative for corrective action.
4	A header retry failure was encountered. The ACI tried four times to read the header and was unable to do so due to LRC errors. Usually this is an indication of bad wiring or incorrect master control software.

TOKEN: /114

FORMAT:

@SAVE

DESCRIPTION:

This commands permits the programmer to save RAPL-II user memory to the EEPROM bank, which is available when the option is installed. Before the @SAVE command is used, the write protect switch on the top of the EEPROM card must be selected to the write enable position. This is done by removing the controller lid, and sliding the switch with a small screwdriver. Once the @SAVE function is completed, slide the switch back to the original write protect setting.

WARNING:

Data loss will result if the write protect switch is not returned to the write protect setting after the @SAVE is completed and before the next time the controller is turned ON.

APPLICABLE MODES:

{I},{M},{P}

EXAMPLES:

1. Save the complete user memory
   >> @SAVE

CROSS-REFERENCES:

@RESTORE, ALLOC, FREE

Error Number	Error Description, Cause, and Recovery
18	Time-out on receiving the SOH character. The master control, after establishing communications with the slave during the enquiry sequence, has a limited amount of time to send the following header block. A time-out error always causes the ACI to be reset.
19	Time-out on receiving the header ETX character. This time-out measures the time it takes to complete the transmission of the header block. Once the SOH has been received, the master has a limited time to send the whole header block. The ACI will be reset after this error.
20	Time-out on receiving the data block STX character. when the header block has been completed, and the master has programmed a READ cycle, the master must transmit the complete block within a specified time.
21	Time out on data block ETX/ETB character. Similar to error #19, the master has a limited amount of time to transmit the entire data block. The ACI will be reset after this error.
22	Time-out on receiving a data block ACK or NAK character. After the slave has transmitted the data block to the master, it expects a response within a certain time limit. The ACI will be reset after this error.
23	Time-out on receiving the final EOT. The slave expects the master to close the communication within a certain time limit. The ACI will be reset after this error.
24	Unexpected EOT character. This error will be set if any expected control characters turns out to be an EOT. The master can close communication at any time by sending an EOT instead of an expected control character.
25	Not used.
26	Bad special read code. The read code provided in the header is not supported.
27	Bad special write code. The write code provided in the header is not supported.
28	Bad special memory type code. The memory type code provided in the header is not supported.

Table B-3 - ACI Error Codes.

TOKEN: /137

FORMAT:
@@RN

DESCRIPTION:

This command selects a slave device number for the robot controller. This value can be any value between 1 and 127. The value of 0 is not permitted.

APPLICABLE MODES:

{I},{P},{M}

B-10 PREPARING THE ROBOT CONTROLLER FOR A COMMUNICATION MODE

The robot controller must be set up to accept the communication. All ACI communication must be through device #1. When the ACI is activated, any attempt to use device #1 as a normal serial communication channel in RAPL-II will result in a RAPL-II error (reserved I/O).

B-11 ACI MONITOR COMMANDS

The programmer can access the ACI software through a set of commands available at the terminal. These commands allow the user to monitor communication to and from the controller and to enable, disable or alter the parameters of the ACI interface.

The RAPL-II password must have been issued before the user may access any of these commands.

@@RI: Forces an initialization of the ACI interface. Any communication in sequence will be aborted. This may cause a loss of synchronization between the master and slave.

@@RN: To select a slave device number for the robot controller. This value may be between 1 and 127. The value 0 is not permitted.

@@RS: Display the current ACI software status. This includes a static display of slave number and whether on or off and a dynamic display showing number of successful; cycles since reset, number aborted, number failed, last error, data block pointer, and several flags. This is a useful debugging tool.

@@RE: Enable the ACI software. This will dedicate device #1 to use as an ACI port. No other serial I/O will be allowed on this device.

@@RD: Disable the ACI software. Device #1 will function as a normal serial channel.

@@RH: Display the contents of the last header received by the slave. Also useful for debugging.

TOKEN: /134

FORMAT:

@@RE

DESCRIPTION:

This command enables the ACI software. This will dedicate the serial port #1 to the ACI software. No other normal serial input will be allowed on this channel. Output is still allowed, but it will cause problems if an ACI cycle is concurrently running.

APPLICABLE MODES:

{I},{M}

42h This command will force a write of the user input port image as captured by the RAPL-II I/O scanning software. No memory address is required in the header block, but the size should be set to 9 bytes. APPENDIX F contains a description of the digital input structure.

43h This command will force a write of the user output port image as set by the RAPL-II I/O scanning software. No memory address is required in the header block, but the size should be set to 9 bytes. APPENDIX F contains a description of the digital output structure.

44h This command will force a write of the position command registers in the robot controller. The position command is an array of 8 double integers in memory. No memory address is required in the header block, but the size should be set to 32 bytes.

45h This command will force a write of the actual position registers in the robot controller. The actual position is an array of 8 double integers in memory. No memory address is required in the header block, but the size should be set to 32 bytes.

46h This command will force a write of the motion end-point registers in the robot controller. The motion end-point position is an array of 8 double integers in memory. No memory address is required in the header block, but the size should be set to 32 bytes.

47h This command will transfer 8086 I/O input port values to the host. The I/O address is specified by the offset portion of the data address field of the header block. The segment field is not used. This is a special command in that only one partial block of data may be transferred. The number of bytes in the last block will correspond to the number of ports to be scanned.

48h The command will turn off the ACI protocol and its exclusive use of DEVICE 1 once the sequence is complete. Terminal mode is started on DEVICE 1 automatically with the transmit enable line set. A sequence of ^Z ^Z characters from the terminal will re-establish ACI and turn off the terminal mode. These two commands allow the ACI and terminal communication to run consecutively on the same serial line.

Special READ Codes

There are currently no special read codes supported.

TOKEN: /122

FORMAT:

@NOA

DESCRIPTION:

This command displays the current number of axis, and permits the user to change the number.

DEFAULT CONDITION

At power up, the controller reads the number of axis cards installed. If there are more or less than expected the system will prompt the user to enter the correct number. The system will not permit further operation until a response has been entered. If this sequence is aborted, the RAPL-II interpreter will be entered, but arm power will not be enabled.

APPLICABLE MODES:

{I},{M}

C-1 ROUTINE MECHANICAL MAINTENANCE SCHEDULE

This section describes the recommended maintenance schedule for the mechanical system and the procedures to be followed during maintenance periods.

The recommended maintenance schedule is found in table C-1.

Maintenance item	Description	Hourly Interval
Service Transmission Chains	page 2	2000
Check Inner Wiring Harness	page 3	2000
Adjust Wrist Gear Mesh	page 4	2000
Check arm covers for wear/ cracking	page 5	2000
Check Motor brush wear	page 6	2000
Inspect clean and refit Harmonic Drives	page 8	8000

TABLE C-1 Scheduled Mechanical Maintenance

TOKEN: /167

FORMAT:

@MAXVEL [axis#],[VELOCITY]

DESCRIPTION:

This command is used to tune the maximum value of command velocity to the maximum pulse output rate of the encoder for an extra axis.

If this value is set improperly, the encoder of the extra axis may not be anble to send out pulses fast enough to keep up with the command. The axis may lag behind so badly that the command error may overflow its buffer and the motor will reverse. In less drastic cases, the motor will be poorly synchronized with the robot, particularly at higher command speeds, and it will be difficult to control the axis under manual control.

Determine the value of [VELOCITY] by running the motor at full speed under the given load and amplifier conditions, and measure the encoder speed in RPM. The encoder output can be used as a tach if a scope can be attached to its output. The frequeny along with the opulse count per revolution will give the speed in RPM using the following formula:

$$\text{ENCODER RPM} = \frac{60}{(\text{Pulses/rev})(\text{Pulse-Period}_{sec})}$$

WARNING:

The @XMAXVEL command must be executed after any @XPULSES or @XRATIO command, since the @XMAXVEL command generates internal parameters that are dependant upon all extra axis parameters.

APPLICABLE MODES:

{I},{M}

C-1.2 WRIST GEARS

A) Wrist Gear Mesh Adjustment:

TOOLS REQUIRED: - allen keys
- small gear puller

PROCEDURE:

1) Remove gripper and any other components mounted to the wrist flange. With power on, put robot at a fixed starting position (READY pose works well). Observe the relative position of the joint 5 homing marks. The mesh of the wrist gears later depends on being able to move the robot to the same pose and install the wrist at the same relative position.

2) Enter the command "LIMP 4" then remove the motors 4 and 5 fuses to "limp" the wrist.

3) Remove the wrist flange: Undo the #8-32 Button-Head Cap Screw at the centre of the wrist flange. Screw a 1" long 8-32 SHCS into the threaded hole in the wrist shaft. Using the small gear puller, pull the flange bearing clear of its seat on the shaft. The screw of the puller should bear on the head of the screw inserted in the shaft. Excessive force from the puller should not be necessary, the flange may slip off with just manual pressure. Ensur that no dirt or metal filings get into the grease on the gear.

4) With the flange removed, two 6-32 SHCS screws will be visible on either side of the wrist flange shaft. The shaft is held axially against a 10-32 set screw by these two screws. Adjusting the axial position of this shaft will change the mesh of the wrist bevel gears. To tighten the mesh, loosen the 4-40 screws by about a quarter of a turn. Flip the wrist by 180 degrees and locate the set-screw opposite to the wrist flange. Loosen the set-screw by about one-half turn. Return to the 6-32 screws and tighten them against the set screw to shift the shaft axially.

5) Install the wrist flange on the shaft. Install the BHSCS and tighten to ensure the bearing is fully seated on the shoulder of the shaft. Since the shaft had moved in axially, the gear mesh should be tighter than normal. Adjust the set screw slowly until the mesh is just right. Then tighten past that by about 1/4 turn.

6) Carefully remove the wrist flange again. Tighten the 6-32 screws carefully and equally to prevent tipping of the shaft. At this point, a quick check of the gear mesh may be done to determine whether further adjustment is required.

7) Install the motor fuses, UN-LIMP the wrist, and move the arm to the starting position. Install the wrist flange and re-align the flange to the same tooth mesh as before by checking the alignment of the home marks. When the flange bearing is seated, re-check the quality of the gear mesh. If not correct, repeat the adjustment procedure steps 4) - 6).

TOKEN: /162

FORMAT:

@LOFB <SET[<Limit>,<Time_out>]|RESET>

DESCRIPTION:

This command sets or resets the Loss Of FeedBack (collision or servo failure) detection check. When set, this function will issue an error for any axis which is being commanded at the programmed limit for more than the programmed time with no change in position. These symptoms are typical of the following situations:

1. Broken encoder, producing no feedback.
2. Blown fuse.
3. Robot has collided with an object and is binding.
4. Static overload.

The values of limits and time-out can be set in A250 controllers only (ie. not in A150 controllers). The units of <Limit> are in pulses while the units of <Time_out> is in milli-seconds.

DEFAULT CONDITION

For version 3.4.5 and 3.4.6, the default condition is off (Reset). For all versions from 3.5.0 and on, and RAPL-II, the default condition is on (Set). The default limit and time values in RAPL-II are 115 pulses and 400 msec.

APPLICABLE MODES:

{I},{M}

EXAMPLE:

1. To activate collision detection

```
>>@LOFB SET<CR>
>>
```

2. To de-activate collision detection

```
>>@LOFB RESET<CR>
>>
```

3. To check current limits and set a higher sensitivity limit:

```
>>@LOFB<CR>
SET
 POSITION ERROR IN PULSES:   +115
 TIME DURATION (MSEC): 00500

>>@LOFB SET,75,400<CR>
>>
```

C-1.3 WRIST DRIVE GEARS

A) Mesh Adjustment

TOOLS REQUIRED: - allen keys
- High-pressure Moly-based gear grease DOW Corning DP2

PROCEDURE

1) Remove arm plastic covers. The two main half covers can be removed by undoing flat-head screws at the front of the skirt, at the top of the shoulders, and at the rear above the rear motor cover. Disconnect the two wrist motors by un-plugging their connectors.

2) Remove Wrist motor plate. Do this by loosening three 10-24 screws below the wrist motors and pulling the plate up and away. This will remove the mesh between the spur gears in the drive train. Leave the three 10-24 screws which hold the plate in position installed.

3) Check the play in the bevel gears. The play may be adjusted by moving the spur-gear/bevel gear cartridge on its shaft. The axial position of the cartridge is determined by the screw in the end of the shaft and a small wave-washer spring between the cartridge and the wrist drive centre block. Adjust the play by turning the screw. Clock-wise will loosen the gear mesh and visa-versa. The screw is located with Loctite at the factory to prevent it from coming out of adjustment. This may result in it being difficult to adjust. If loctite has gotten on the end of the shaft, the outer bearing may be tight. In this case, carefully remove and clean the parts before adjusting.

4) If the mesh of the gears is still somewhat rough when adjusted, check for burrs or debris in gear teeth. If the rougness persists, it may be necessary to shim the motor to adjust the pinion/gear squareness.

5) Repeat the procedure for both the wrist motor/gear drives as required.

6) Liberally apply high pressure gear grease (DOW Corning DP2 or equivalent) onto the bevel gear and pinion. Before re-installing the motor plate to the robot, apply this grease to the spur gear pinion as well.

7) Install wrist drive subassembly on the 10-24 SHCS's, meshing the spur gears loosely. Move the assembly around to find the best mesh of both sets of gears. When it feels as if the gears are properly meshed, gently tighten the centre screw (of the three holding the assembly). Move the wrist up and down with your hand. Is there excessive play on the gear mesh; is the mesh too tight; are the gears aligned properly? Both sound and feel will indicate if the gears are set up properly. By trial and error adjust the mesh. When the gear sets are adjusted satisfactorily, tighten the other 2 screws holding the assembly on the plate. Move the wrist again to determine if the gears are still adjusted properly. Repeat if necessary.

8) Re-install the plastic covers.

TOKEN: /178

FORMAT:

@INIT <AXES|AXIS|DARTA|DARTB|PIC|PIT|VARS|LOCS|PROGS|ALL>

DESCRIPTION:

This command initializes a part of the controller hardware. This should only be done in the event of a failure, and under direction of a qualified service technician.

THIS COMMAND IS NOT FOR GENERAL USE.

AXES or **AXIS** selection will initialize all or one selected smart axis card. **VARS**, **LOCS**, **PROGS** will initialize the appropriate table. **ALL** will initialize all three tables. **DARTA**, **DARTB**, **PIC**, **PIT** initialize specific hardware components on the mother board.

APPLICABLE MODES:

{I},{M},{P}

CROSS-REFERENCES:

C-1.5 HARMONIC DRIVES

TOOLS REQUIRED:
- Allen keys.
- Long handled tap (6-32 thread).
- Circular Spline puller (CRS Special tool)
- Flex spline puller (CRS Special tool)
- Harmonic Drive grease

PROCEDURE:

NOTE: This repair to be performed only by a factory trained service representative

1) Remove the motor by undoing the four #10 socket cap screws in the mounting flange. Slide the motor carefully out - do not force it. If resistance is encountered, rotating the robot joint while pulling gently on the motor will aid its release.

2) Tap two circular spline holes and remove the spline using the puller (see Figure C-1). Clean all grease from the teeth of the spline.

3) Tap two flex spline holes and remove the flex spline using the puller (see Figure C-2). Clean grease from the inner and outer surfaces and the teeth.

4) Inspect the teeth for signs of wear or damage. A polished or burnished look is normal but wear or deformation of the teeth should be seen as a problem and the drive should be replaced. Inspect the sides of the flex-spline body carefully for any cracks or dents. Some burnishing of the inner surface is normal on the surface where the wave generator runs.

5) Clean the grease away from the wave generator on the motor shaft. Inspect the outer surface of the bearing race for signs of wear or cracks. Remove the snap ring and take out and clean the oldham coupling. Inspect the mating surfaces for wear and check the coupling mesh for play. Grease the coupling parts prior to re-assembly.

6) Before installing the flex and circular splines, ensure that no burrs or foriegn material are trapped under the mounting flanges. Anything which "tips" either spline will cause extremely rough operation after re-assembly and must be avoided.

7) Grease the harmonic drives before completing the installation. Pack the bearing of the wave-generator and spread a film on its outer surface. Also spread a film on the inner surface of the flex-spline and leave a small amount in the center of the flex-spline.

   After both splines are installed, and before installing the motor, the spline teeth may be grease through the grease ports stop injecting grease as soon as it is seen oozing out between the spline teeth. If a grease gun is not available, pack grease behind the circular spline before installing the flex-spline. Also fill the flex-spline teeth with grease before installing it.

TOKEN: /180

FORMAT:

@GANTRY

NOTE:
THIS COMMAND IS USED WITH A150/A250 ROBOTS ONLY. IF THIS COMMAND IS USED WITH ANY OTHER ROBOT CONFIGURATION, AN ERROR WILL BE ISSUED.

DESCRIPTION:

This command sets up a GANTRY condition. This means that every cartesian location includes the position of axes 6 and 7 as X and Y gantry coordinates.

The coordinates of the gantry are stored as a real numbers in 8 bytes immediately above the standard six coordinates of each cartesian entry in the location table. The real values are the "joint" engineering unit values of the axes when the HERE command is issued. Use of @XRATIO, @XLIMITS, and @XPULSES for both axes 6 and 7 is mandatory before issuing this command. Before storing values or commanding motion after the GANTRY command is issued, an XHOME command must have been issued for both axes.

The coordinates stored for the extra axes are entered only as an input to the motion command. Shifting of these coordinates is possible using the SHIFTA command. Entry of a point "off-line" using the POINT command will permit entry of the axis values. All motion commands referring to cartesian locations after issuing the GANTRY command will coordinate with the GANTRY axes.

For long moves involving the gantry axes it may be an advantage to use the SLEW mode. This is entered with the ENABLE SLEW command.

GANTRY is reset using the "TRACK RESET" command.

APPLICABLE MODES:

{I},{M},{P}

CROSS-REFERENCES:

RAPL-II COMMANDS: TRACK, @XPULSES, @XRATIO, @XLIMITS, SHIFTA

OTHER CRS Plus Publications: Technical Manual APPENDIX J

C-2 CLEANING

C-2.1 FILTER CLEANING

The A100/200 Series uses filters to ensure a clean supply of cooling air for the controller. These must be checked periodically and cleaned when necessary. There are two filters: one on the inlet by the fan and the other at the outlet. The inlet filter is designed to be easily removable as it will get dirty fastest. The other may be cleaned less frequently, mostly by just blowing compressed air through it from the inside.

PROCEDURE:

1) Place the controller on a bench, or if the controller is drawer-slide mounted, pull the drawer out fully. Disconnect power from the controller at the rear AC voltage module. Remove the controller lid.

2) The front panel of the controller can be hinged down to access the forward fan filter. To do this, undo and remove the three 10-24 button-head cap screws on each of the sides at the front of the controller. DO NOT remove the 8-32 BHCS screws at the bottom of the controller side panels at the front. The front panel will now hinge down. It will not swing too far as it is mechanically limited from doing so.

3) Once the front panel is out of the way, locate the filter on the left side of the controller behind the fan. Slide the filter forward out of its bracket to remove it.

4) Clean the filter with a solvent and dry thoroughly. Spray it lightly with filter oil or light machine oil before re-installing it.

5) Blow air through the rear filter from the inside out.

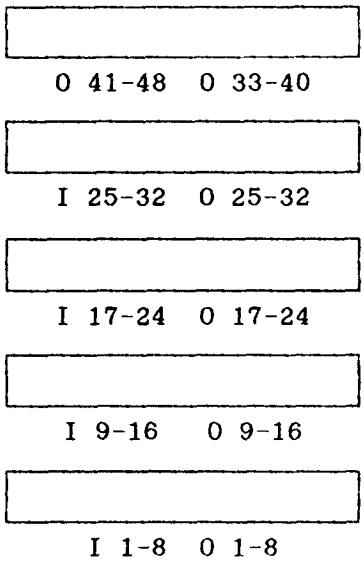
6) Close the front panel, re-install the lid and install the controller as it was.

C-2.2 CLEANING EXPOSED SURFACES

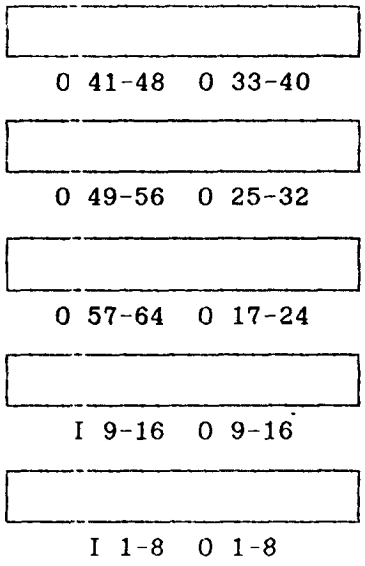
All controller panel and arm components can be cleaned using house-hold cleaning products. The use of solvents or de-greasers may damage the printed surfaces. The plastic covers of the robot arm are made from ABS Plastic. Any cleaner suitable for this material may be used.

When cleaning controller panels, ensure that no liquids are permitted to get into the controller box. Do not submerse any part of the system in any liquids.

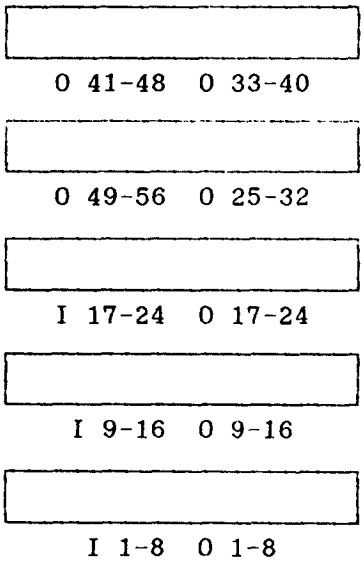
MODE 6



MODE 8



MODE 7



APPLICABLE MODES:
{I},{M},{P}

C-3.2 ENCODER ALIGNMENT

A) Introduction

If a lack of feedback has been diagnosed and it has been determined that neither wiring nor connectors are to blame, the fault may lie with the encoder.

The positional encoder is a precision optical device which translates rotary motion into a string of directional pulses. The actual output from the encoders in the A100/200 Series robot is a pair of square waves which are resolved into CW or CCW pulses by gating circuitry on the axis card. In addition, a zero-crossing pulse (also called the marker-pulse or index-pulse) is produced once for each rotation of the encoder. This latter is only used during homing of the arm.

The square wave is generated as follows:

A metal or glass disk, with a grating on its periphery, is attached to the motor shaft. As the disk rotates, it passes by a static grating (the stator), engraved with the same size lines. A light source shines through the disk and stator gratings to a photo-cell below. As the grating lines alternately line up and oppose, the light level seen by the photo-cell alternates from strong to weak. The more "lines" on the disk, the finer the resolution of the encoder. Circuitry in the encoder amplifies and buffers the varying signal from the photo-cell into a square-wave.

The encoders used in the A100/200 Series arm are Motion Control Device MCD-M21 This is referred to as a "modular" encoder. The concept of the modular encoder is that the disk is contained completely within the stator and the stator/rotor gap is set at assembly by installing the encoder as a whole and then removing a shim. The encoders in the motors of the robot have 1000 lines per revolution.

B) Axial Adjustment of the Disk Relative to the Stator

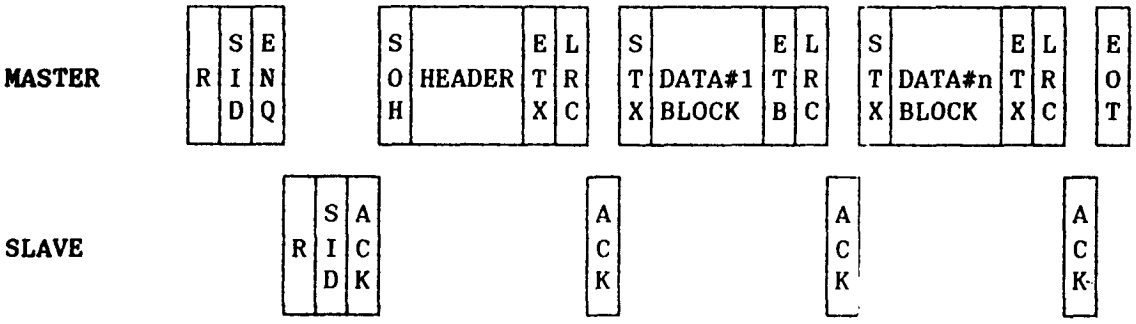
This adjusts the optical gap which is critical for operation. If the gap is too large, the light from the source is not "focused" through the grating and the signal is lost, resulting in a loss of feedback. If the gap is too small, the loading on the shaft, while small, may be enough to cause the two to come into contact, leading to damage of the grating or disk failure.

To adjust the gap, remove the encoder cover and loosen the disk on the shaft. Insert a plastic shim of 0.005" thickness between the disk and stator. Push the disk gently against the shim, trapping it between the disk and the stator. Once the disk is set at the correct position, and tighten screws in the disk hub. Remove the shim and rotate the disk. Look in where the shim was to check for axial run-out of the disk. The gap must not close to closer than 0.003" or there is danger that it may touch the stator under collision loads. If run-out is more than 0.002 TIR, replace the encoder.

This procedure should be followed before any other adjustment of the encoder is attempted.

B-5 DATA BLOCK TRANSMISSION SEQUENCE (Continued)

Similarly in a READ command, the slave will expect blocks of data after the header is acknowledged. It will issue a **NAK** if it detects an error in transmission. The master will issue an **EOT** after the last data block is read and acknowledged:



Data Transfer Error Detection

As in the header block transfer, the LRC byte for a data block is the sum of all data bytes in the block. The **STX**, **ETX** or **ETB** control codes are not included in the summation. An erroneous LRC check will cause the receiving end to issue a **NAK** character instead of the **ACK** which will force a re-transmission of the entire block. Only three retries are permitted before the receiving end issues an **EOT** character to abort the sequence.

B-6 EOT SEQUENCE

The EOT is the final sequence, and is shown in the timing charts above. In the communication, the master ALWAYS has the last word and the master always has the ability to abort the communication. The slave will never knowingly abort the communication once it has been started.

Error Number	Error Description, Cause, and Recovery
5	No ETX or ETB was read at the end of a data block transfer. The ACI will immediately reset after detecting this error.
6	Not used.
7	Not used.
8	Data block read retry failure. The ACI terminated due to repeated LRC failures when reading a data block from the master.
9	Character time-out. The ACI expected a character and did not receive one in the allotted time. The ACI will be reset. Check that the master did not "hang-up" in the middle of the transmission.
10	The ACI read a character other than an STX at the start of a data block. The ACI will be reset after this error.
11	Data block write retry failure. The ACI was unable to successfully send a block to the master without receiving a NAK four times in succession.
12	A character other than an ACK or a NAK was read in response to a data block write. This could be due to excessive noise on the line or incorrect software at the master. The ACI will be reset after this error.
13	Time-out on transmission of a character. The ACI was unable to transmit a character in the allotted time. This time out could be caused by a failure in the cable or the DART chip to provide a transmit ready signal. The ACI will be reset after this error.
14	Bad Target ID match in the header block. The header block must contain the same slave ID as was in the enquiry sequence or this error will be issued. The ACI will be reset after this error.
15	Not used.
16	Bad character received when an EOT expected. The ACI expects the master to close all communication with an ECT character. This error is issued when any other character is received in its place. Since communication should be finished by this time anyway, the ACI will be reset after this error.
17	A character other than an ETX was received at the end of a header block. The ETX must be received immediately after the LRC character.

APPENDIX D - MECHANICAL ASSEMBLY DRAWINGS

Event	Time-out (in character times)
Enquiry Time-out Time-out used to test for characters received during the enquiry sequence.	300
SOH Time-out Time permitted between reception of a successful enquiry sequence, and the acceptance of the header SOH character.	300
Header ETX Time-out Time permitted between acceptance of the header SOH and the detection of the header ETX character.	300
ACK for data block Time permitted for waiting for an ACK or NAK from the master after transmission of a data block from the slave.	300
STX data block Time-out Time permitted for waiting for the data block STX from the master after a successful acknowledgement of either the header block or the previous data block.	300
Data block ETX/ETB Time-out Time permitted for transmission of the complete data block from acceptance of the STX character to reception of either the ETX or ETB character.	300
Final EOT Time-out Time permitted between the acknowledgement of the last data block READ and the final EOT response, or the last data block WRITE acknowledgement from the master, and its response to the final EOT from the slave.	300

Table B-4 - ACI Time-out values

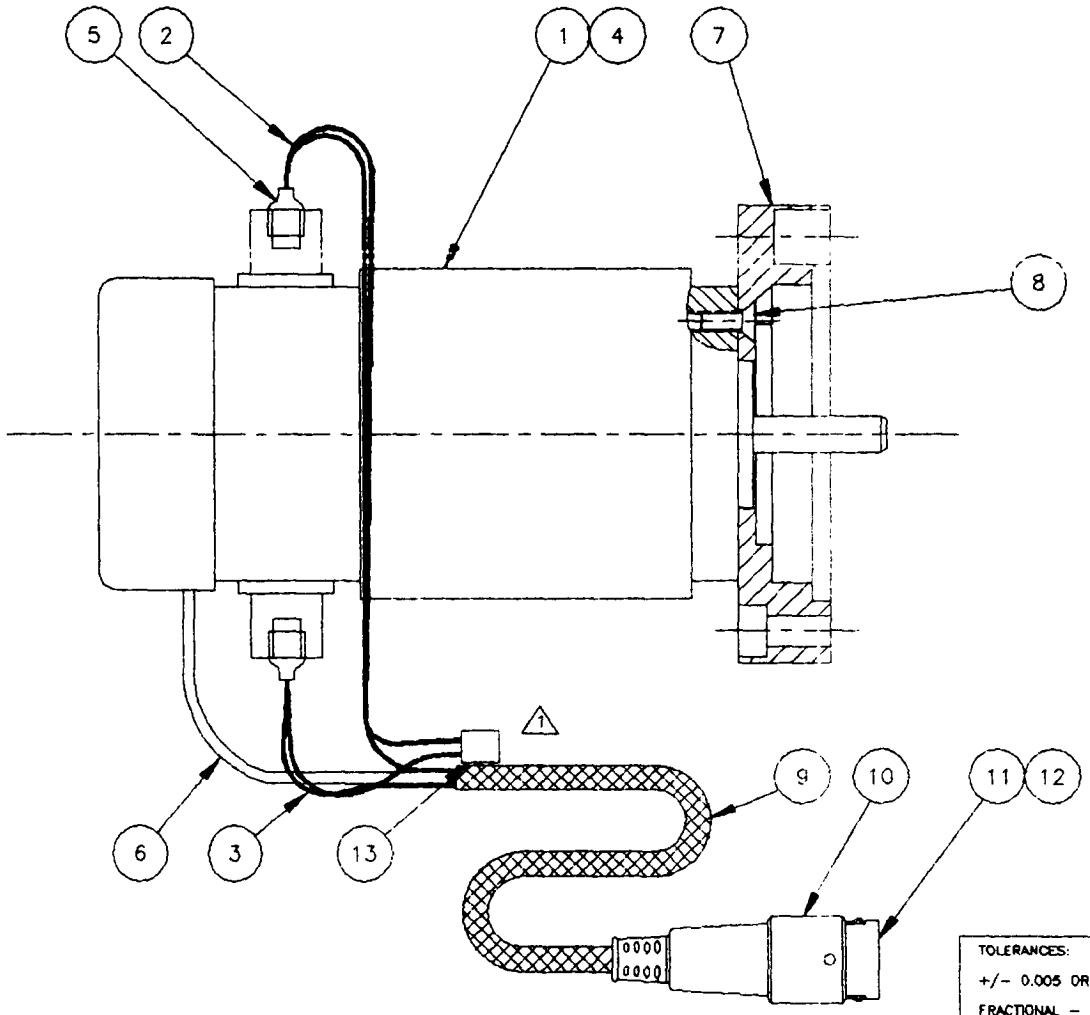
B-8 ABORTING A COMMUNICATION CYCLE

Sending an eot at any time that a normal control character is to be issued will be interpreted by the slave as a signal to abort the communication. This function may be required at times where an abrupt termination of the link is essential, for instance when attempting to quickly service another slave.

B-9 INTERFACE REQUIREMENTS

The ACI has been tested at baud rates up to and including 2400 baud. Faster rates may be possible, but are not recommended as reliability suffers. Handshaking should be disabled, and parity checking turned off. A data byte size of 8 bits is required with one stop bit. Refer to the description of the RAPL-II CONFIG command in the RAPL-II user manual for details on setting up the ACI device (device #1) for the ACI link.

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.
1	OCT 3/90	ADDED CAPACITOR TO MOTOR WIRING	1.0



ITEM	REQ'D	PART NUMBER
13	1	R-CAM103R050 CAPACITOR
12	2	R-CTR/INP-18 INSERT PINS
11	5	R-CTR/INP-24 INSERT PINS
10	1	R-CTR/12P-RC ROUND CONNECTOR PLUG
9	.88'	R-WSV/04NYLN 1/4 GP NYLON BRAID
8	4	R-F0632FHC06 FLAT HEAD SCREWS
7	1	T-ABK-M91413 H/D FLANGE
6	.33'	R-WHS/03BKXX HEAT SHRINK TUBING
5	2	R-WSL/03SPAD SPADE LUG
4	1	R-WCT/12D060 WIRE TIE
3	1'	R-WHU/18RDXX WIRE
2	1'	R-WHU/18BKXX WIRE
1	1	R-MTR/ME2110 DC SERVO MOTOR

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH.

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 2/3 MOTOR ASS'Y

ASS'Y No: S-SMC-14-027

	DATE	NAME
DRAWN	DEC. 14, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SB-14-003

PART No: S-SMC-14-003

B-12 SPECIAL ACI CODES

This section describes the special read and write codes that are available with the ACI software. These special codes make it easy for the programmer to obtain specific information that would otherwise be cumbersome.

The special codes are separated into special WRITE codes (code 40h to 4Fh) and special READ codes (codes 20h to 2Fh). These codes are placed in the MEMORY TYPE specifier section of the header block.

Special WRITE Codes

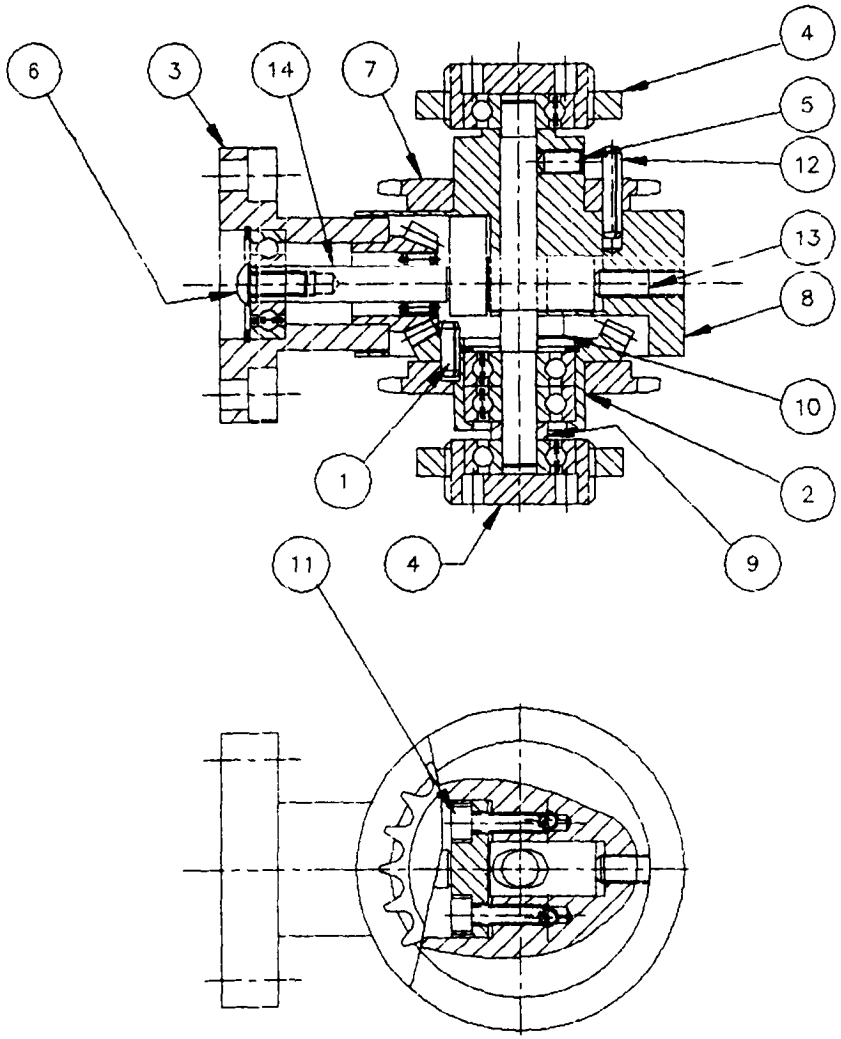
With the exception of code 47h, it is not necessary to specify an address with special write commands. It is however important in all cases to specify the amount of memory to be transferred.

40h This special write will return the pointer table, a list of relevant addresses within the robot software. The use of this function will permit the master to operate independently from the robot software version. For example, the pointer table contains the addresses of all components of the user memory, permitting up- and down-loading of programs, locations, variables, calibration, etc.

The buffer of information which is returned is in the form of memory pointers (4-byte, segment:offset format). The lowest byte contains the least significant byte of the pointer. The addresses included in the list of information point to the items in robot memory as described in table G-1 from APPENDIX G of the A100/200 Technical Manual. To determine the size of memory required, look for the relevant item in the list and multiply the offset of that item, plus one, by 4. Bear in mind that this special transfer always starts with the first element in the list.

41h Send Robot Error codes. This code will cause the robot to send four bytes to the master indicating the alarm (error) status of the controller. The four bytes, in order, are:

1	ALARM STATUS	00	No ERROR condition.
		01	An error which has not been cleared exists.
2	LAST RAPL-II ERROR #	nn	See RAPL-II Manual APPENDIX B
3	ACI ERROR	00	No ACI ERROR condition
		01	An ACI error which has not been cleared exists.
4	LAST ACI ERROR #	nn	See ACI ERROR list.



REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.

ITEM	REQ'D	PART NUMBER
14	1	R-M/C-M09107 FLANGE AXLE
13	1	R-F1032CHS06 10-32x3/8 SET SCREW
12	2	R-F02RLPIN10 1/8Dx5/8 ROLL PIN
11	2	R-F0632SHC06 6-32x3/8 SHCS
10	2	R-F0632SHC04 6-32x1/4 SHCS
9	1	R-M/C-M09127 AXLE SPACER
8	1	T-ABK-M09117 WRIST CTR. BLOCK
7	1	R-M/C-M09116 WRIST SPROCKET
6	1	R-F0832BHC04 8-32x1/4 BHCS
5	1	R-F0832CHS04 8-32x1/4 SET SCREW
4	2	S-SMC-09-036 BEAR'G ADJUSTER ASS'Y
3	1	S-SMC-09-035 FLANGE SUB-ASS'Y
2	1	S-SMC-09-034 BEVEL GEAR ASS'Y
1	1	R-F02RLPIN06 1/8Dx3/8 ROLL PIN

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: WRIST ASSEMBLY VER. 2

ASS'Y No: SMC-14-037

	DATE	NAME
DRAWN	DEC. 5, 1989	P. D. Young
CHECKED		

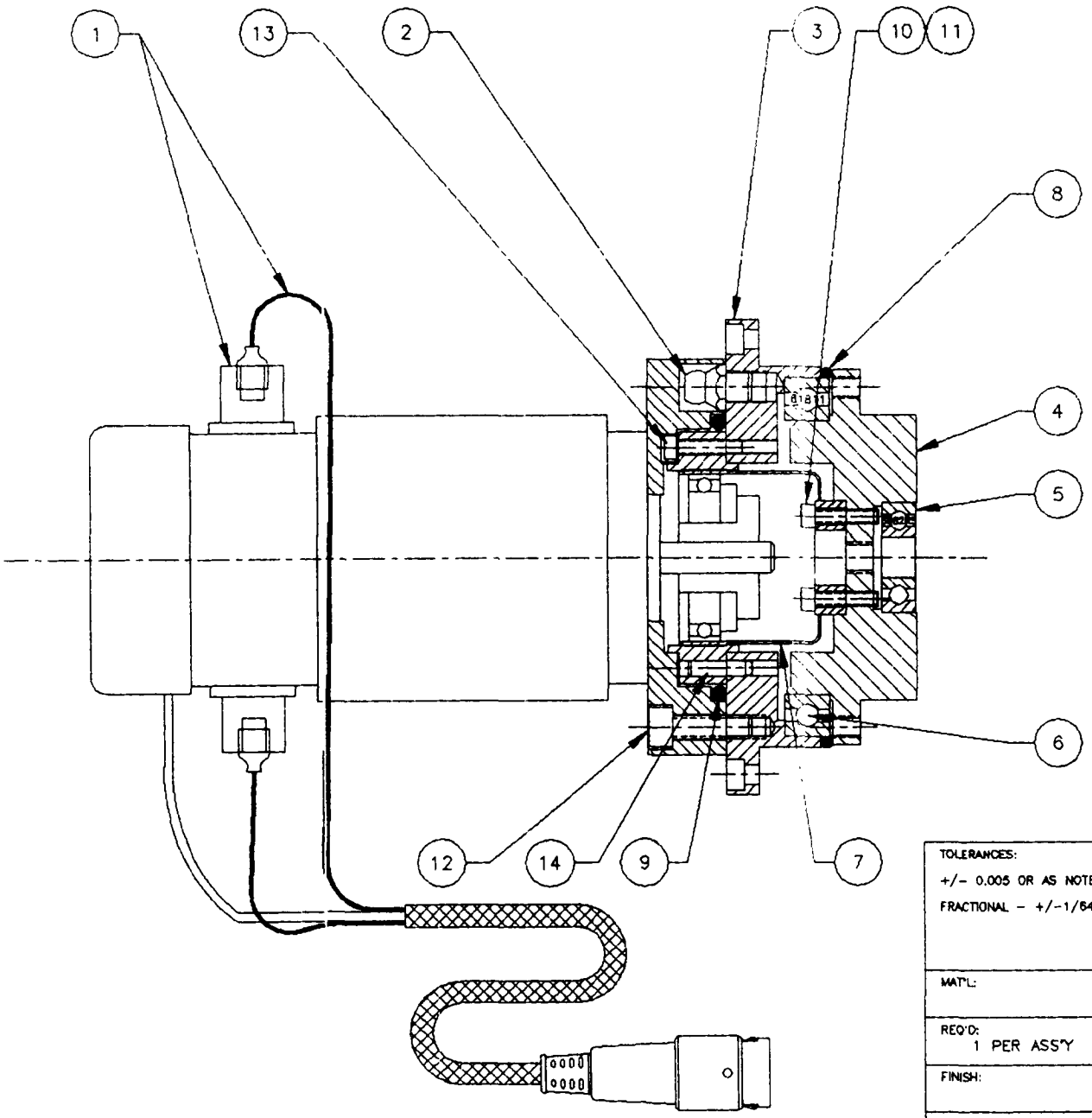
DWG No: SB-14-009

PART No: S-SMC-14-009

SCALE: FULL SIZE

This APPENDIX contains information on routine maintenance, cleaning and some mechanical adjustments which may be required during the service life of the robot.

TABLE OF CONTENTS



REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.

ITEM	REQ'D	PART NUMBER
14	2	R-F02DLPIN10 DOWEL PIN
13	6	R-F0440SHC10 CAP SCREW
12	4	R-F1024SHC10 CAP SCREW
11	2	R-F02DLPIN08 DOWEL PIN
10	6	R-F0440SHC08 CAP SCREW
9	1	R-GRO-RNG21D O-RING SEAL
8	1	R-GRO-RNG2.5 O-RING SEAL
7	1	R-TRC/1C072A HARMONIC DRIVE
6	1	R-BG00061811 BEARING
5	1	R-BG0608-2RS BEARING
4	1	R-M/C-M91403 JOINT 2 FLANGE
3	1	T-ABK-M91402 SHOULDER FLANGE
2	1	R-GRF/LI5010 GREASE FITTING
1	1	S-SMC-914-003 JOINT 2/3 MOTOR ASSY

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 2 DRIVE ASSEMBLY

ASS'Y No: S-RSA-14-003

	DATE	NAME
DRAWN	DEC. 4, 1989	P. D. Young
CHECKED		

DWG No: SB-14-027

PART No: S-SMC-14-027

C-1.1 DRIVE CHAINS - ADJUSTMENT AND LUBRICATION

TOOLS REQUIRED: - Long allen key driver - 3/32" AF
- #0 Robertson screwdriver
- Loctite #601 retaining compound
- Heavy molybdenum-based oil

PROCEDURE:

1) With robot arm power on, check chain tension on the slack side of the loop when the robot is carrying its expected load. This can be tested easily at the inner side of the elbow joint. Play in the chain is obvious when there is a "dead" feel in the slack (unloaded) side of the chain when pressed from side to side. If a dead feel can be detected, procede to adjust the tension as described below.

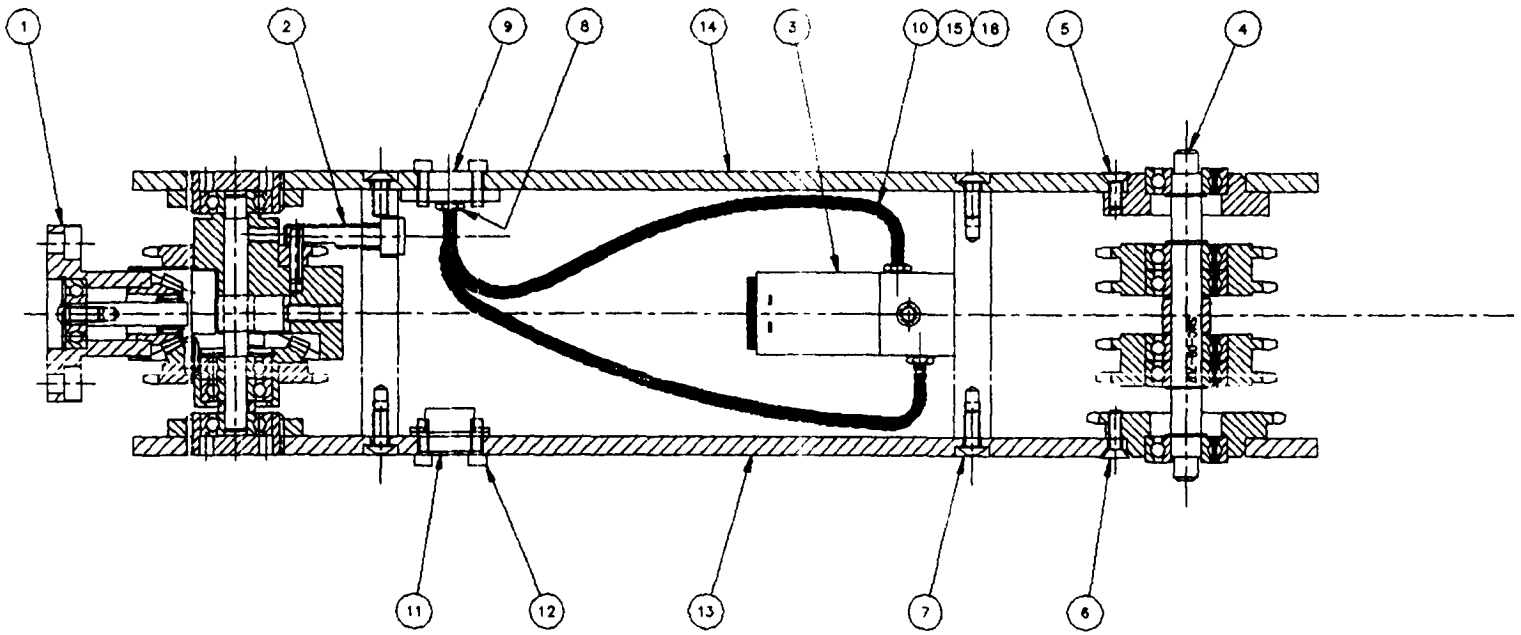
2) Remove rear lower arm cover, take out all screws on top section of upper arm cover and bend it back to expose the insides of the upper arm. This will give access to all five of the chain tensioners.

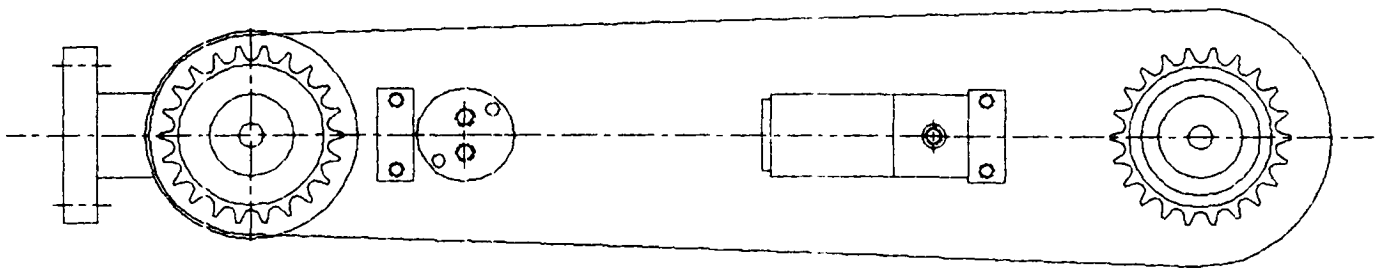
3) Using the long handled 3/32" allen key tighten the screws in the tensioner until moderate finger pressure fails to deflect the chain as described above. It should not take much to do this. Do not overtighten as damage to the chain, sprocket and joint bearings could result.

4) After the chain has been tightened, apply a small drop of Loctite medium strength retaining compound (green colour) to each screw in the tensioner. Use sparingly and ensure that none gets on the chain.

5) Apply a molybdenum compound heavy weight oil to the chain. Use sparingly. Because of the slow speeds the chain must endure, this should be sufficient lubrication. However, under severe operating conditions, a moly grease may be more suitable.

REVISION			
ITEM	DATE	DES.	REL.





ITEM	REQ'D	PART NUMBER
16	2	R-WMP/25AB25 CABLE TIE PLATFORM
15	5	R-WCT/12D060 WIRE TIE
14	1	T-SWH0-09180 LINK2 RH SIDE PLATE
13	1	T-SWH0-09100 LINK 2 LH SIDE PLATE
12	4	R-F0440SHC06 SH CAPSCREW
11	1	R-CTR/12S-RC HIROSHI RECEPTACLE
10	1.16'	R-PNH/SL02BK 1/8" AIR HOSE
9	1	T-ABK-M09156 AIR BULKHEAD
8	2	R-PNF4B10-32 BARB AIR FITTING
7	8	R-F0832BHC04 BUTTON HEAD CAP SCREW
6	3	R-F0632FHC08 FH CAP SCREW
5	3	R-F0832FHC06 FH CAP SCREW
4	1	S-SMC-14-707 JOINT 3 SHAFT ASSY
3	1	S-SMC-09-033 AIR VALVE ASS'Y
2	1	S-SMC-09-045 J4 HARD STOP ASS'Y
1	1	S-SMC-09-009 WRIST ASS'Y

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

CRSPLUS
INDUSTRIAL AUTOMATION

850 Harrington Court,
P.O. Box 153, Station A
Burlington, Ontario
L7N 3N4

MATL:

REQ'D: 1 PER ASS'Y

FINISH:

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: LINK 2 SUBASSEMBLY

ASSY No: S-SMC-14-044

	DATE	NAME
DRAWN	DEC. 12, 1988	P. D. Young
CHECKED		

DWG No: SC-14-037

PART No: S-SMC-14-037

B) Wrist Gear Lubrication:

TOOLS REQUIRED: - allen keys
- small gear puller
- High-pressure Moly-based gear grease DOW Corning DP-2

PROCEDURE:

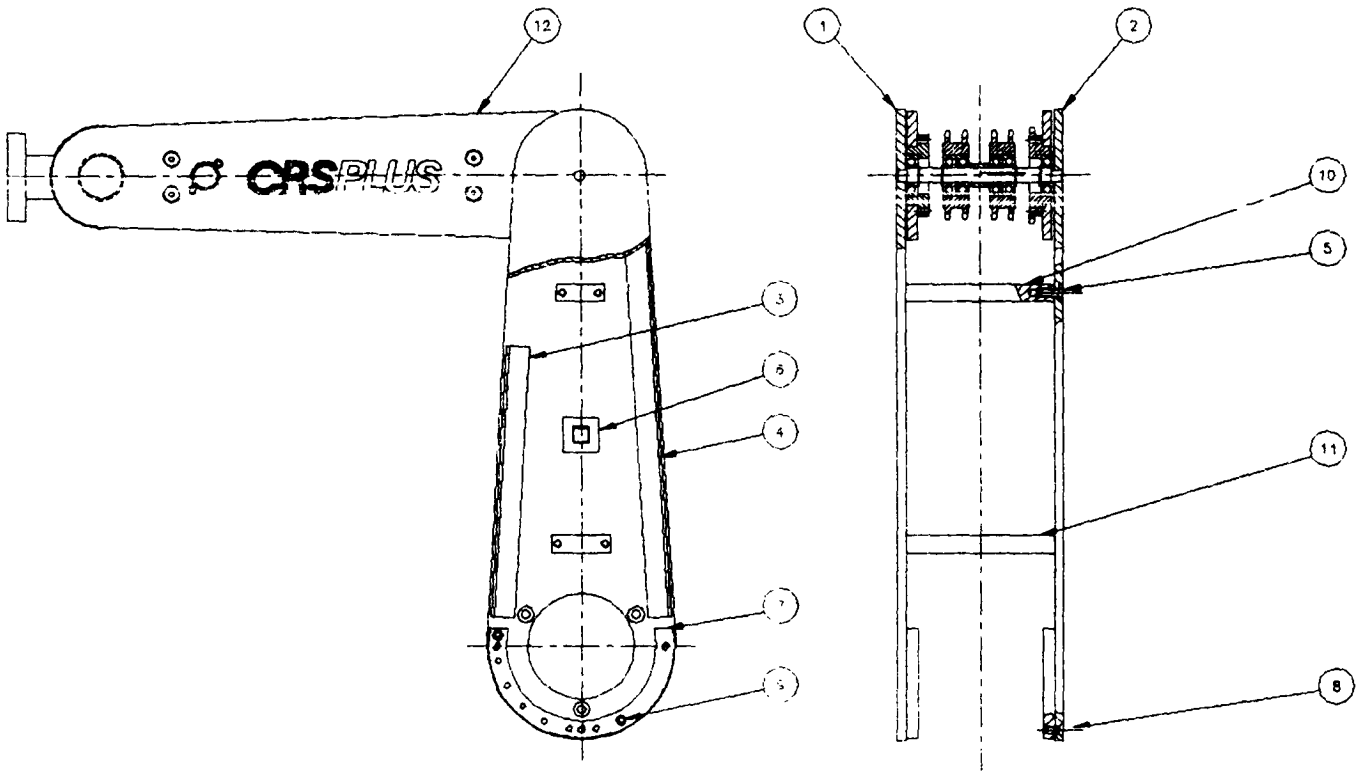
1) Remove gripper and any other components mounted to the wrist flange. With power on, put robot at a fixed starting position (READY pose works well). Observe the relative position of the joint 5 homing marks. The mesh of the wrist gears later depends on being able to move the robot to the same pose and install the wrist at the same relative position.

2) Remove the wrist flange: Undo the #8-32 Button-Head Cap Screw at the centre of the wrist flange. Insert and bottom out a 1" long 8-32 SHCS into the threaded hole in the wrist shaft. Using a small gear puller, pull the flange bearing clear of its seat on the shaft. The screw of the puller should bear on the head of the screw inserted in the shaft. Excessive force from the puller should not be necessary, the flange may slip off with just manual pressure. At all times, ensure that no dirt or metal filings gets into the existing grease on the gears.

3) Apply High-pressure molybdenum disulphide based gear grease such as DOW Corning DP2 to the pinion gear pressed into the flange. Pack the same grease into the small needle bearing in the pinion. Use a small clean screwdriver to ensure that the grease gets into the needles of the bearing. Remove any excess.

4) Re-install the flange so that the gears mesh loosly. LIMP the wrist and rotate the flange by hand to distribute the grease on the bevel gear. UN-LIMP the arm and re-position the robot at the starting position. Remove the flange again and apply a bit more grease to the pinion. Install the flange again. This time ensure that the gears mesh properly as noted by the position of the joint 5 homing marks. Tighten the cap screw to approximately 18 Kg-cm.

REVISION				
ITEM	DATE	DE	TION	REL



ITEM	REQ'D	PART NUMBER
12	1	S-SMC-09-037 LINK 2 SUBASSEMBLY
11	1	R-M/C-M09175 LINK 1 CROSSMEMBER B
10	1	R-M/C-M09028 LINK 1 CROSSMEMBER A
9	2	R-F0632SHC04 STOP SCREWS
8	6	R-F0832FHC06 FLAT CAP SCREW
7	2	T-ABK-M91408 STOP RING
6	3	R-WMP/25AB25 CABLE MOUNT PLATFORM
5	8	R-F0832BHC04 BUTTON HEAD SCREWS
4	1	R-M/C-M91419 LONG FLEX CVR GUIDE
3	2	R-M/C-M91416 SHORT FLEX CVR. GUIDE
2	1	T-PWH-M91417 LINK 1 RH SIDE PLATE
1	1	T-PWH-M91409 LINK 1 LH SIDE PLATE

PART LIST

TOLERANCES
+/- 0.005 OR AS NOTED
FRACTIONAL = +/-1/64

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 153 Station A
Burlington, Ontario
L7N 3N4

MATL:	PROJECT: A150/250 ROBOT ARM		JOB No. CRS-914
REQ'D: 1 PER ASSY	TITLE: ARM SUBASSEMBLY		ASSY No. S-RSA-14-003
FINISH		DATE / NAME	DWG No. SC-14-044
	DRAWN	DEC. 21, 1988 / P. D. Young	
	CHECKED		PART No. S-SMC-14-044
	SCALE	FULL SIZE	

C-1.4 MOTOR BRUSHES

TOOLS REQUIRED: - 1/4" slot screwdriver
- 1/4" offset slot screwdriver

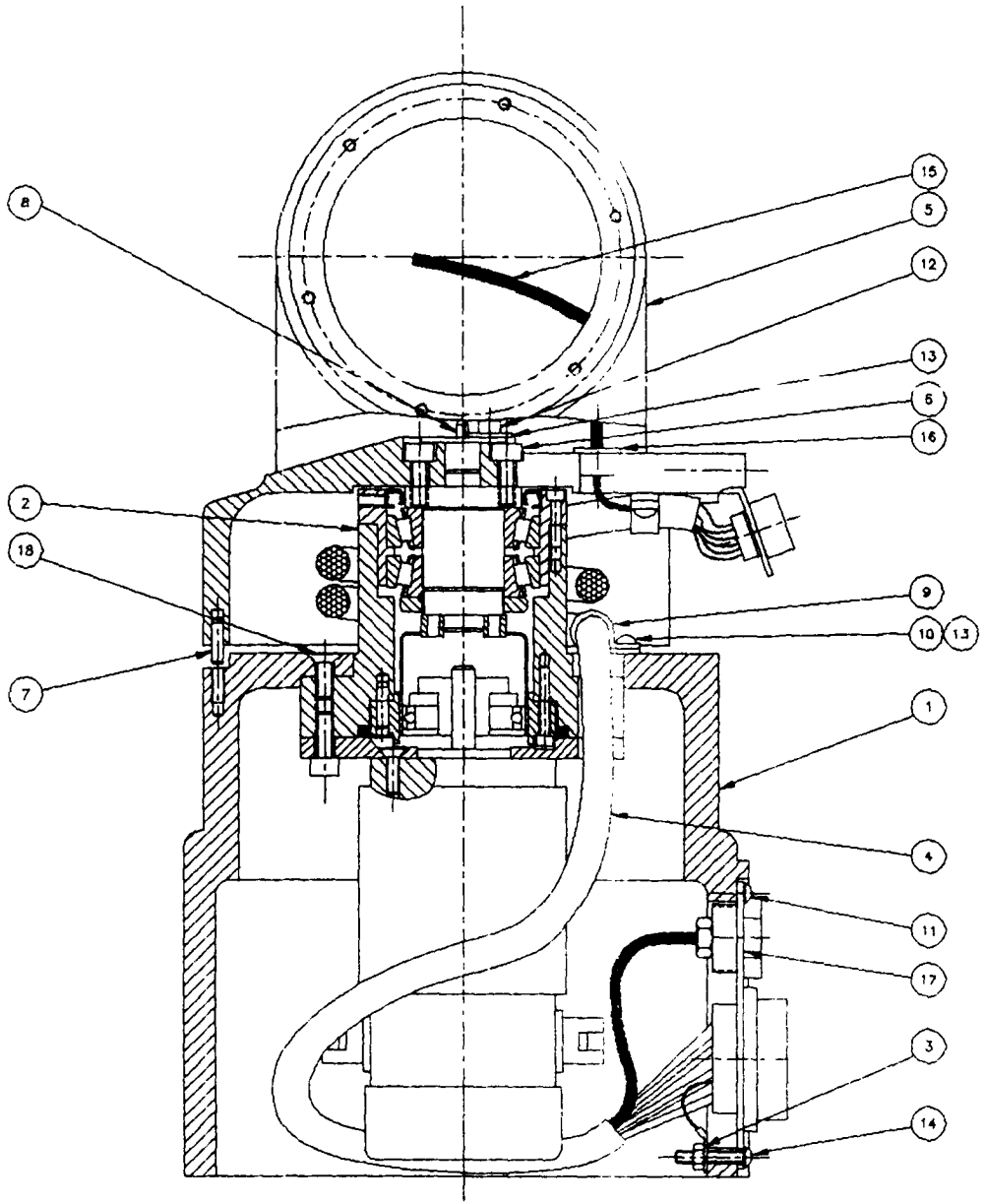
PROCEDURE:

9) The brushes for each motor are located on the outside of the rear bell-housing of each unit. The only motor which will give accessability problems is the joint 1 motor. This one is located inside the base casting. Its brushes are accessable with an offset screwdriver.

10) The brushes should be removed, inspected, and replaced one at a time to prevent re-assembling them incorrectly. Putting the wrong brush into the housing will result in the motor running away uncontrollably when power is applied. This would be devastating and must be avoided!

11) The Brushes should be routinely changed every 10000 hours or if the brush length is less than 0.19 inches from the seat of the spring to the tip.

REVISION TA			
ITEM	DATE	DESCR... ION	REL



ITEM	REQ'D	PART NUMBER
18	4	R-F1024FHC08 FLAT HEAD CAP SCREWS
17	1	R-ZSN/M1A-RA ARM S/N PLATE
16	1	R-WCC/06BU04 WIRE BUSHING 3/8ø
15	33	R-WHS/03BKXX HEAT SHRINK TUBING
14	1	R-F0632RRM12 MACHINE SCREW
13	4	R-F08FLTWASH #8 FLAT WASHER
12	2	R-F0632SHCS04 CAP SCREW
11	5	R-F0632RRM04 MACHINE SCREW
10	2	R-F0632RRM08 MACHINE SCREW
9	2	R-WCC/08ID08 CABLE CLAMP
8	1	R-F02RLPIN16 ROLL PIN
7	2	R-F02DLPIN08 DOWEL PIN
6	4	R-F1024SHC08 CAP SCREWS
5	1	T-PBL-M91401 WAIST PAINTED BLACK
4	1	S-SMC-14-043 MAIN HARNESS ASSEMBLY
3	1	R-F0632HEXNT 6-32 HEX NUT
2	1	S-SMC-14-106 NECK ASSEMBLY
1	1	T-PBL-M91411 BASE PAINTED BLACK

PART LIST

TOLERANCES
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASSY

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

520 Huntington Court
P.O. Box 163 Station A
Burlington, Ontario
L7R 3W4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: BASE SUBASSEMBLY

ASSY No: S-RSA-14-003

	DATE	NAME
DRAWN	DEC. 18, 1998	P. D. Young
CHECKED		

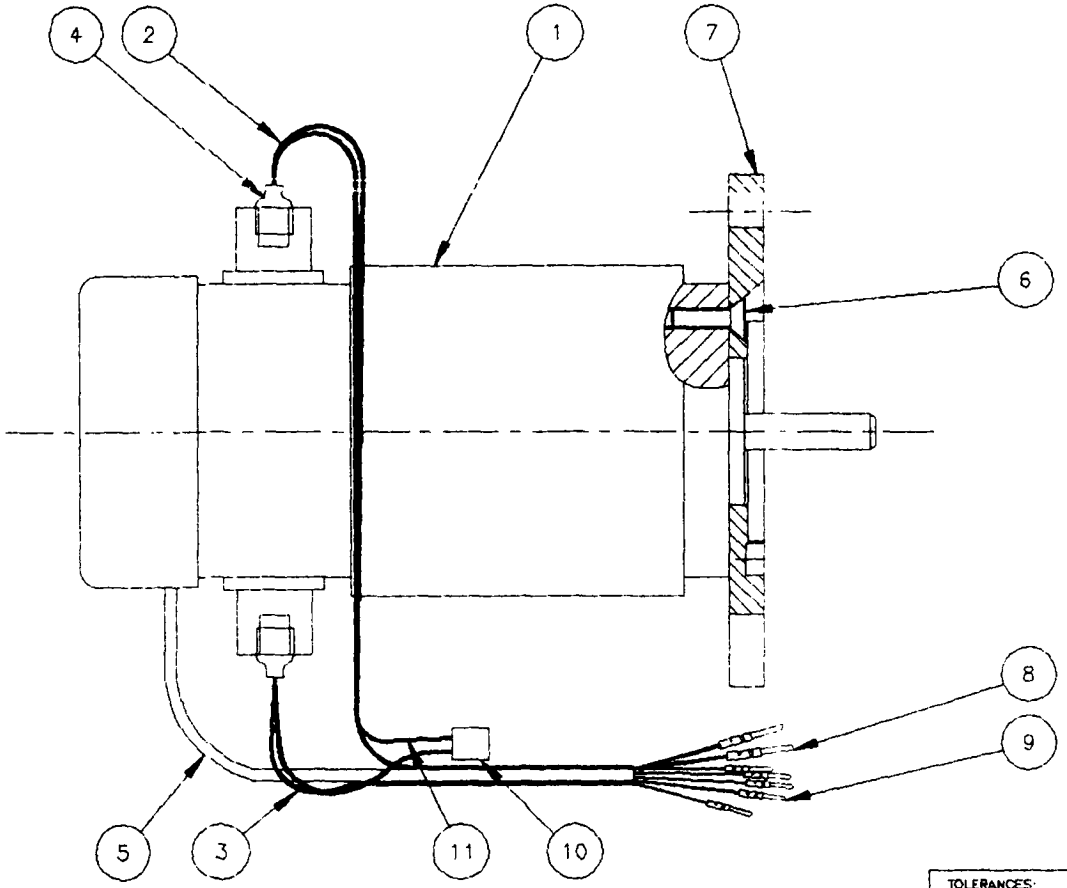
DWG No: SC-14-104

PART No: S-SMC-14-104

SCALE: FULL SIZE

8) Re-calibrate the robot and check the homing marks. Dis-assembly of the drives may change the relative location of the encoder index in which case the homing marks will need to be changed.

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
11	.2'	R-WHS/01BKXX HEAT SHRINK TUBING
10	1	R-CAM103R050 CAPACITOR
9	5	R-CPC/CCS505 AMP 24 GA SOCKET INS.
8	2	R-CPC/CSS101 AMP 18 GA SOCKET INS.
7	1	T-ABK-M09224 H/D FLANGE
6	4	R-F0632FHC08 FLAT-HEAD SCREWS
5	.33'	R-WHS/03BKXX HEAT SHRINK TUBING
4	2	R-WSL/03SPAD SPADE LUG
3	.8'	R-WHU/18RDXX WIRE
2	.8'	R-WHU/18BKXX WIRE
1	1	R-MTR/ME2110 DC SERVO MOTOR/ENCDR.

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: MOTOR 1 ASSEMBLY

ASS'Y No: S-SMC-14-106

	DATE	NAME
DRAWN	DEC. 19, 1989	P. D. Young
CHECKED		

DWG No: SB-14-107

PART No: S-SMC-14-107

SCALE: FULL SIZE

C-3 MECHANICAL ADJUSTMENT PROCEDURES

C-3.1 WRIST CENTERING

TOOLS REQUIRED: - Hook Spanner
- Pin wrench

PROCEDURE:

1) Remove the screws holding the top surface of the upper arm cover in place. Pull back the cover to expose the wrist assembly.

2) Measure the distance between the wrist centre block and the inner edge of the arm side plate on both sides of the wrist. If the difference from one side to the other is less than 0.005 inches there is no need to perform an adjustment.

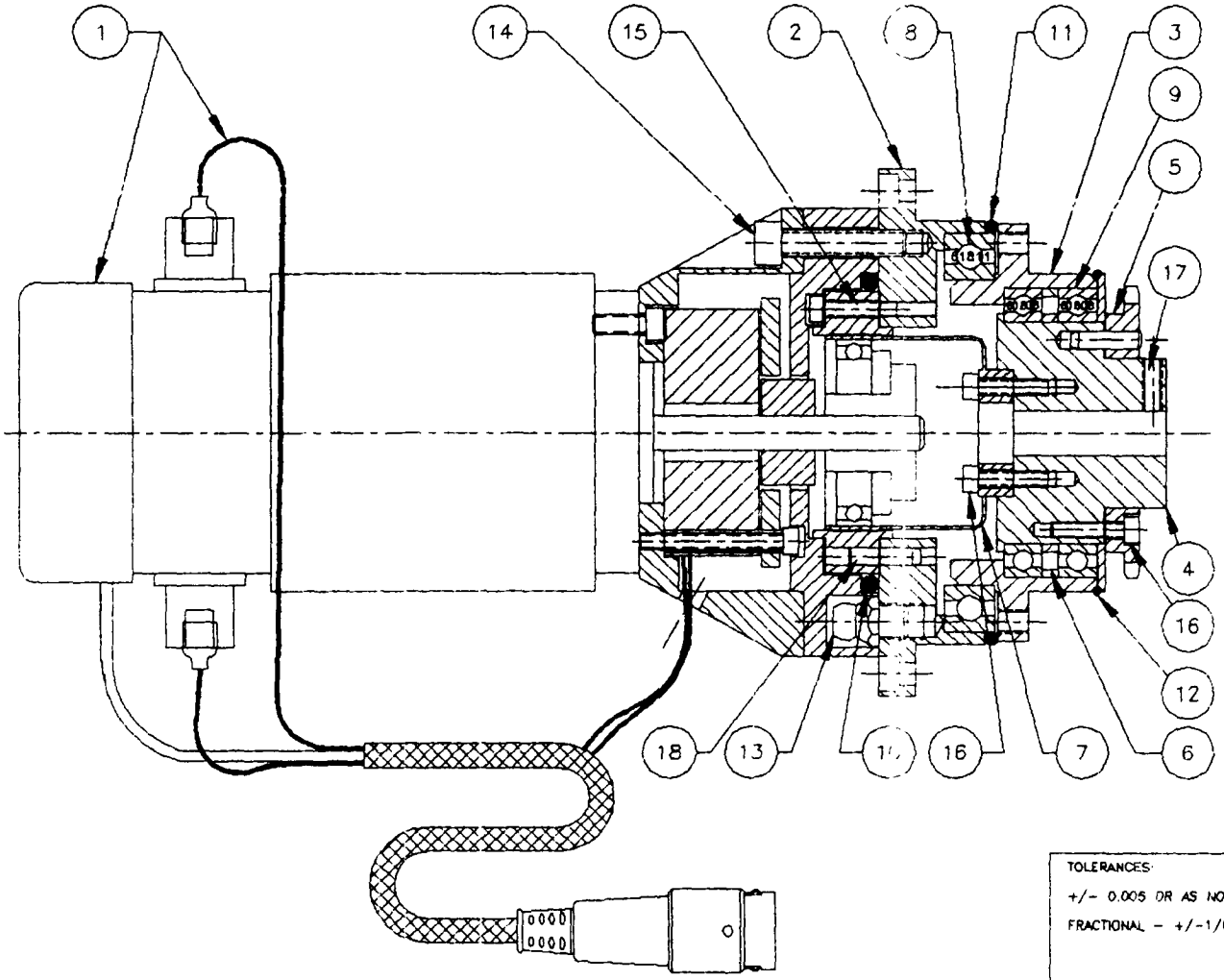
3) To adjust the spacing, use the hook spanner or other suitable tool to loosen the lock nuts holding the wrist cross axle bearing adjusters in place.

4) Use the pin wrench to first loosen the wrist bearing adjuster which is on the side of the arm with the larger gap to the wrist block. Loosen no more than 1 turn at a time.

5) Use the pin wrench to tighten the wrist bearing adjuster which is on the side of the arm with the smaller gap to the wrist block. Tighten just enough to take up the distance left by loosening the other side.

6) Repeat the above steps until the difference from side to side is less than 0.005 inches. When complete, tighten the locking nuts. Replace the arm cover.

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
18	4	R-F02DLPIN08 DOWEL PIN
17	2	R-F0632CHS06 SET SCREW
16	10	R-F0440SHC08 CAP SCREW
15	6	R-F0440SHC10 CAP SCREW
14	4	R-F1024SHC14 CAP SCREW
13	1	R-GRF/LI5010 GREASE FITTING
12	1	R-GRO-RNG2.1 O-RING SEAL
11	1	R-GRO-RNG2.5 O-RING SEAL
10	1	R-GRO-RNG2ID O-RING SEAL
9	2	R-BG00061608 BEARING
8	1	R-BG00061811 BEARING
7	1	R-TRC/1C072A HARMONIC DRIVE
6	1	R-M/C-M91407 J3 BEARING SPACERS
5	1	R-M/C-M91410 JT 3 DRIVE SPROCKET
4	1	R-M/C-M91405 JOINT 3 PIVOT
3	1	T-ABK-M91404 J3 FLANGE
2	1	T-ABK-M91402 SHOULDER FLANGE
1	1	S-SMC-14-103 JOINT 2/3 MOTOR ASS'Y

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A151/251 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 3 DRIVE ASS'Y

ASS'Y No: S-RSA-14-103

	DATE	NAME
DRAWN	DEC. 15, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SB-14-129

PART No: S-SMC-14-129

C) Centering of the Stator Relative to the Shaft

If the stator is not centered about the shaft, the gratings will not be correctly aligned. If the mis-alignment is severe enough, the signal will be lost.

Centering of the stator is best done with an oscilloscope which will permit monitoring of the two signals A and B coming from the encoder. Without a proper fixture, this is the only way to tell if the optics of the encoder are aligned.

Turn the robot main power on without the arm power. Set the scope to look at the signals present on the feedback Molex on the mother board for the axis in question. To set the stator position, slightly loosen the two screws holding the stator to the rear of the motor. Hold the stator tightly against the motor housing to maintain the correct gap and spin the motor shaft while watching the signals from the scope. Move the stator until two strong square wave signals can be seen at the motherboard input. Tighten the encoder down and re-check the signal. Repeat if necessary.

APPENDIX E - OPTICAL ENCODER SPECIFICATIONS

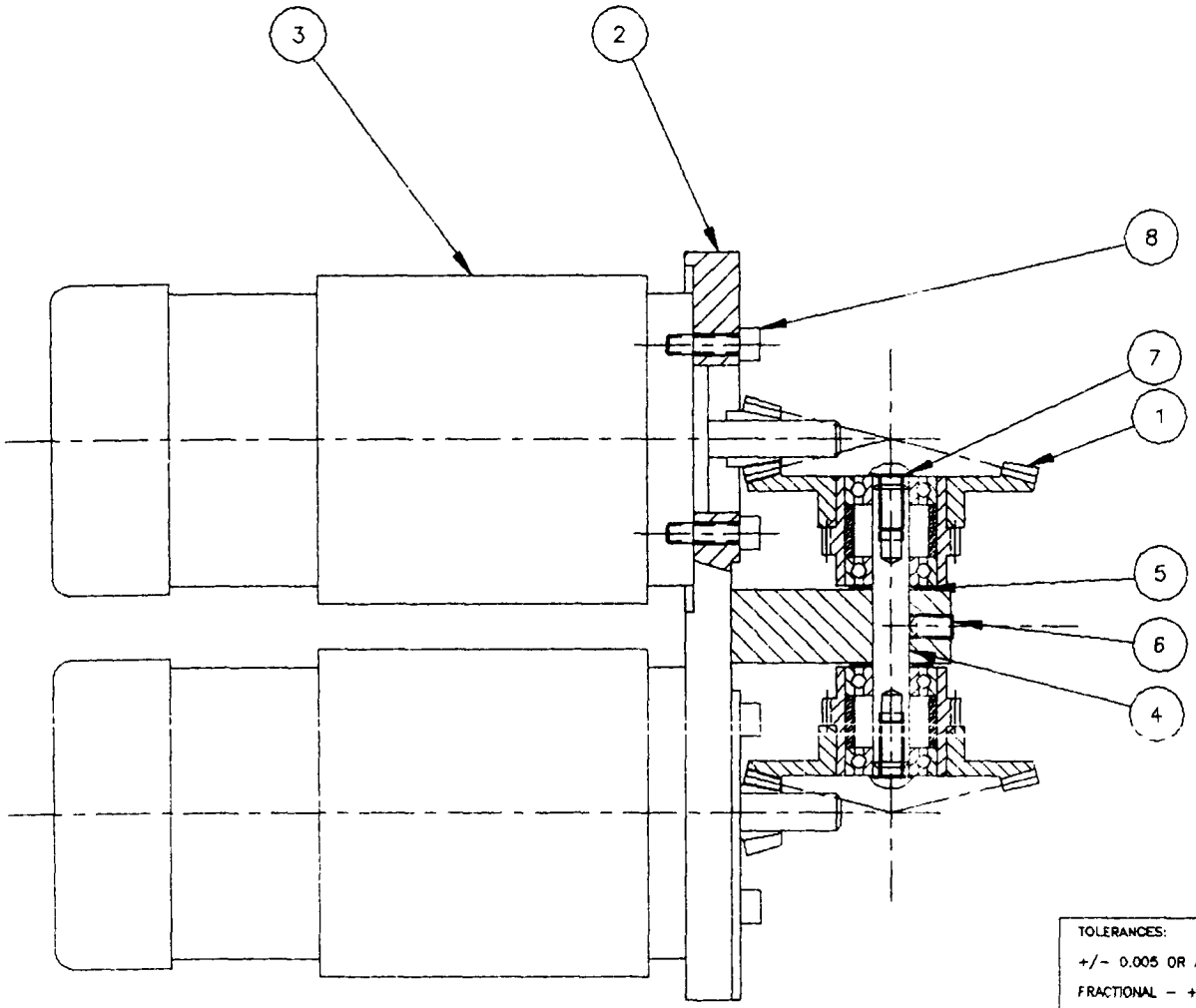
The A100/200 controller can be interfaced to optical encoders with the following specifications:

- 2-channel (A, B) output with zero-cross signal or 4-channel (A, A*, B, B*) with zero-cross (Z, Z*).
- Square wave output.
- 5 volt DC power supply input with <120 milliamp current typ.
- Signal rise time of 1.0 microsecond max.
- Signal fall of 1.0 microsecond max.
- Pulse frequency range of DC to 50 kHz max.

APPENDIX F - CONTROL SYSTEM TECHNICAL DESCRIPTION

[This APPENDIX contains detailed descriptions of the operation and archi-tecture of the SRS-M1A Controller. It is intended for those who require specialized interfacing or plan operating software modifications. For access to this document, please contact your distributor or CRS Plus.]

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
8	8	R-F0632SHC08 CAP SCREW
7	2	R-F0832BHC04 BUTTON HEAD CAP SCREW
6	1	R-F0832CHS04 SET SCREW
5	2	R-F04WVEWASH WAVE WASHER
4	1	R-M/C-M09152 WR. MOTOR GEAR SHAFT
3	2	S-SMC-09-040 WRIST MOTOR ASS'Y
2	1	S-SMC-09-039 WR. MOTOR PLATE ASS'Y
1	2	S-SMC-09-038 SPUR-GEAR/BEVEL ASS'Y

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-009

TITLE: WRIST DRIVE ASS'Y

ASS'Y No: S-RSA-09-003

	DATE	NAME
DRAWN	DEC. 19, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

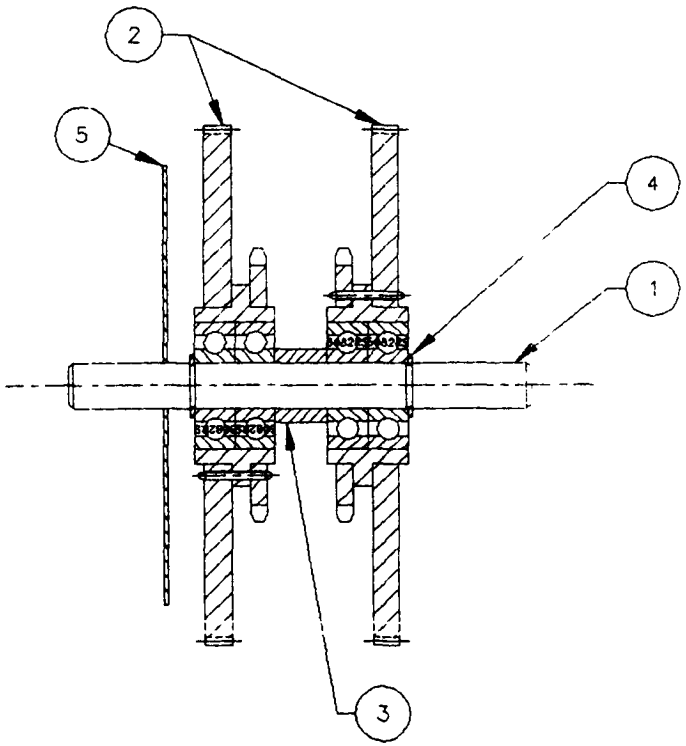
DWG No: SB-09-002

PART No: S-SMC-09-002

APPENDIX G - RAPL-2 CONTROL PARAMETER LIST

TABLE OF CONTENTS

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
5	1	R-M/C-M09181 PROTECTIVE DISK
4	2	R-BGR-EX0312 SNAP RING
3	1	R-M/C-M09027 BUSHING
2	2	S-SMC-09-025 SPUR GEAR/SPRCT ASS'Y
1	1	R-M/C-M91406 JOINT 2 SHAFT

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 2 SHAFT ASS'Y

ASS'Y No: S-RSA-14-003

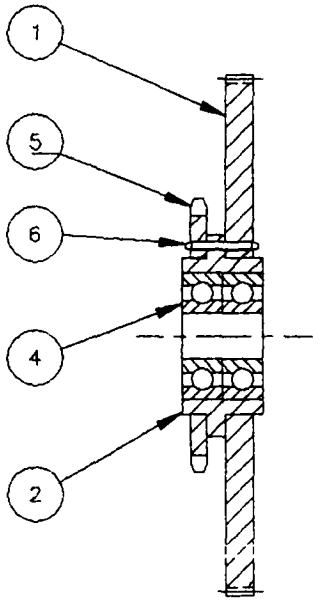
	DATE	NAME
DRAWN	DEC. 4, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SB-[illegible]4-006

PART No: S-SMC-14-006

Data Type	Offset in List	Item Description (This pointer points to..)
W	4	Program table size - the Word register which tells us how long the program table is in number of total program entry locations. **PROG_TABLE_SIZE**
P	5	Variable table pointer - the Pointer to the variable table. **VAR_TABLE_PTR**
W	6	Variable table size - Word register which tells us how big the variable table is in the total number of entries. **VAR_TABLE_SIZE**
P	7	Location table pointer - Pointer to the location table. **LOC_TABLE_PTR**
W	8	Location table size - Word register which tells us how long the location table is in the total number of table entries. **LOC_TABLE_SIZE**
DI*8	9	Calibration register pointer - Pointer to an array of 8 double integers which contain the robot calibration position. **CALIBRATION(8)**
B	10	Calibrate position checksum byte - The byte addition of all valid calibration data for the 5 robot axes only. **CALIBRATE_CHECKSUM**
B	11	'robot is calibrated' - check byte which is set after a calibration procedure has been run. **ROBOT_IS_CALIBRATED**
B	12	**TRACK$SPEC** register - Register contains the state of the extra axes, and whether they are included in the cartesian Xfrm or not.
-	13	**NULL**
DI*8	14	Actual Position array - An array of 8 double integers which are the absolute position registers of the robot feedback sub-system. Every 4 milliseconds, these values are updated with the incremental positions determined by the motor encoders. **POSITION**
DI*8	15	Position Command array - An array of 8 double integers which represent the absolute position command to the robot motors. Any path generation algorithm can create these values, and the servo loop function will compare these registers to the actual position registers and will create the appropriate commands. **POSITION_COMMAND**

REVISION TABLE		
ITEM	DATE	DESCRIPTION



ITEM	REQ'D	PART NUMBER
5	2	R-F01RLPIN08 ROLL PIN
4	2	R-BG0608-2RS BEARING
3	1	R-M/C-M09146 JT 2 WRIST SPROCKET
2	1	R-M/C-M09150 JT 2 WRIST DRIVE HUB
1	1	R-M/C-M09149 LARGE SPUR GEAR

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
2 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: SPUR GEAR/SPROCKET ASS'Y

ASS'Y No: S-SMC-14-006

	DATE	NAME
DRAWN	JULY 25, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SB-[illegible]4-025

PART No: S-SMC-14-025

Data Type	Offset in List	Item Description (This pointer points to..)

R*8 24 Acceleration array - An array of 8 reals which determine the acceleration of each motor during a joint interpolated motion. **ACCELERATION(8)**

R*8 25 Transmission Ratio array - An array of 8 real numbers which equates the measurement units of the joint or motor with the corresponding motor pulse resolution. It is dangerous to change the first five values in this array, as it will change the robot transformation. The @XRATIO command changes the other three.

B 26 Alarm Number - A byte value that contains the last RAPL error code to be generated. **ALARM_NUMBER**

W*8 27 Rotary Resolution array - An array of 8 integers which specifies the motor encoder resolution in encoder lines per motor revolution. **ROTARY_RESOLUTION(8)**

R*6 28 Tool Transform - an array of 6 real numbers that specify the current tool transform value. **TOOL_TRANSFORM**

Struct 29 Input block structure - the RAPL character input buffer area. The input buffers are specified in the following format:

```
input_block(3) structure (
     buffsize byte,
in_ptr byte,
out_ptr byte,
buffer(128) byte)
```

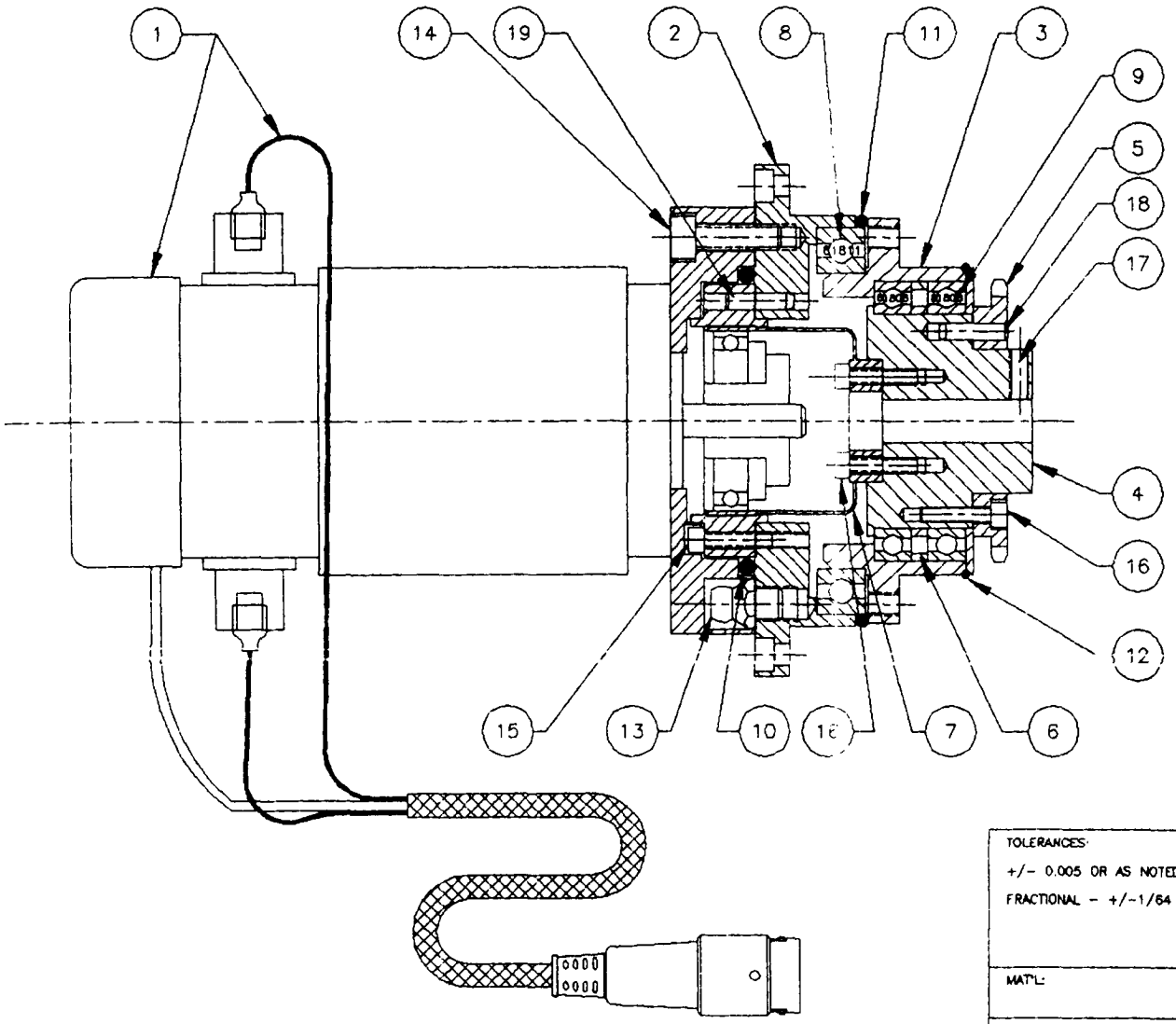
The first two structure records keep track of data input from the two serial ports. The third is used as an intermediate buffer when executing from a program. 'buffsize' determines the size of the input ring buffer, up to 128 bytes. 'in_ptr' points to the next array element available for serial input, and 'out_ptr' marks the next character that can be released from the buffer, to be used by RAPL.

B 30 Alarm status - A boolean byte value which specifies whether or not a RAPL error has occurred. The byte is reset when a new command is attempted. **ALARM_STATUS**

R*8 31 Motor to Joint transform factor array - A real array which transforms the motor coordinates into the joint coordinates.

- 32 **NULL**

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
19	1	R-F02DLPIN10 DOWEL PIN
18	3	R-F02DLPIN08 DOWEL PIN
17	2	R-F0632CHS06 SET SCREW
16	10	R-F0440SHC08 CAP SCREW
15	6	R-F0440SHC10 CAP SCREW
14	4	R-F1024SHC10 CAP SCREW
13	1	R-GRF/LI5010 GREASE FITTING
12	1	R-GRO-RNG2.1 O-RING SEAL
11	1	R-GRO-RNG2.5 O-RING SEAL
10	1	R-GRO-RNG2ID O-RING SEAL
9	2	R-BG00061608 BEARING
8	1	R-BG00061811 BEARING
7	1	R-TRC/1C072A HARMONIC DRIVE
6	1	R-M/C-M91407 J3 BEARING SPACERS
5	1	R-M/C-M91410 JT 3 DRIVE SPROCKET
4	1	R-M/C-M91405 JOINT 3 PIVOT
3	1	T-ABK-M91404 J3 FLANGE
2	1	T-ABK-M91402 SHOULDER FLANGE
1	1	S-SMC-14-003 JT 2/3 MOTOR ASS'Y

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 3 DRIVE ASS'Y

ASS'Y No: S-RSA-14-003

	DATE	NAME
DRAWN	DEC. 4, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SP -14-029

PART No: S-SMC-14-029

Data Type	Offset in List	Item Description (This pointer points to..)

B 42 Teach name count - The byte value which is encoded into the final three character positions of the teach name when the point is finally stored. **TEACH_NAME_COUNT**

B 43 Program pause flag - When a program is executing, setting this byte to a '1' value will stop the program after the next line is decoded. **PROGRAM_PAUSE**

B 44 Homing complete flag - This flag is set when the robot has been homed after a power up. Do not confuse this with item 44, which indicates whether or not the robot has been calibrated. This flag must be set in order that all move functions that access points will work. **ROBOT_IS_HOMED**

B*128 45 String buffer - The string buffer is a buffer 128 bytes long, that is divided evenly into 4 32 byte areas. These correspond to the 4 string variables that are allowed by RAPL version 3.50 or later. **STRING_BUFFER**

B 46 Loss of feedback/collision detection byte - This byte indicates the status of this function. A boolean true means that the function is turned on. **LOFB_CHECK**

B 47 Program completed flag - indicates whether or not a program is completed. **PROGRAM_COMPLETED**

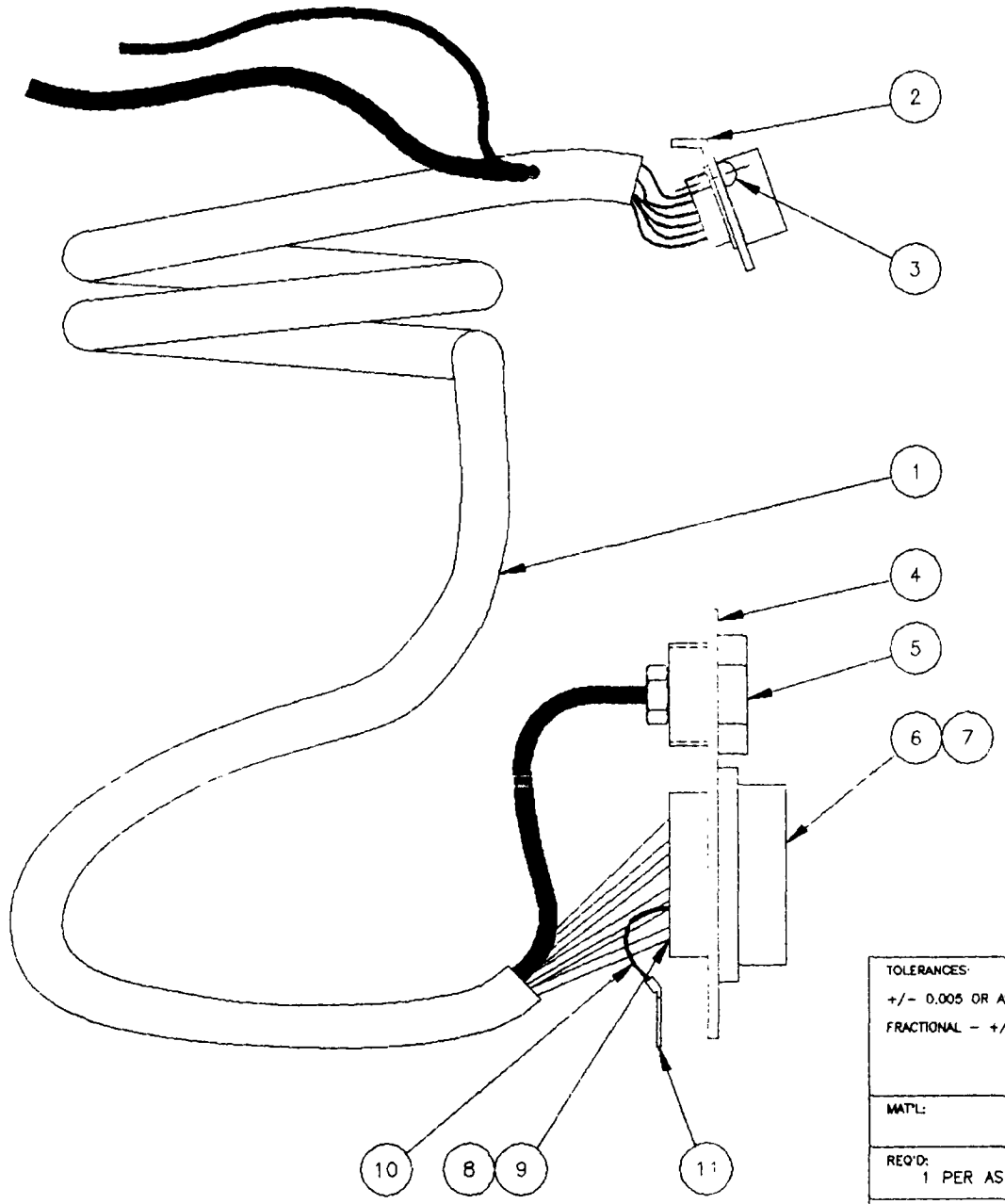
W 48 Program repetitions counter - word counter that indicates the number of repetitions that the current program has completed. **PROGRAM_REPETITIONS**

W 49 Program repetitions requested - the word register which indicates the number of repetitions of the program that have been requested. **PROGRAM_REPEAT_LIMIT**

B 50 Loop forever Flag - the flag which indicates that an infinite looping of the current program has been requested. **PROGRAM_LOOP_FOREVER**

R*250 51 Straight-line Buffer - the 1000 byte buffer which contains the data for the current straight line move. Also used to contain the path parameters for a CPATH. **STRAIGHT_LINE_BUFFER**

B 52 Trigger Enabled flag - flag which indiactes whether or not the trigger table has been enabled. **TRIGGER_TABLE**



REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.

ITEM	REQ'D	PART NUMBER
11	1	R-WSL/06RING #6 RING TERMINAL
10	.33'	R-WHU/18GNXX GROUND WIRE
9	12	R-CPC/CCS099 AMP 18 GA SOCKET INS.
8	26	R-CPC/CCS505 AMP 24 GA SOCKET INS.
7	1	R-CPC/SFR438 57 PIN AMP RECEPTACLE
6	1	R-CPC/SFR043 14-PIN AMP RECEPTACLE
5	1	S-SMC-09-042 AIR FITTING ASS'Y
4	1	T-ABK-M91412 BASE CONNECTOR PLATE
3	8	R-FM2.5RSM05 MACHINE SCREW
2	1	T-PBK-M91417 RP-17 PLATE, BLACK
1	1	S-SCC-09-518 MAIN ARM HARNESS

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: MAIN HARNESS SUBASSEMBLY

ASS'Y No: S-SMC-14-104

	DATE	NAME
DRAWN	DEC. 20, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SP-14-043

PART No: S-SMC-14-043

Data Type	Offset in List	Item Description (This pointer points to..)

W 66 Size of the CTPATH buffer - Pointer to the word parameter which contains the number of bytes allocated for the path buffer. **PATH_BUFFER_SIZE**

R 67 CTPATH Timer - **TIMER** which increments as the continuous path proceeds. This timer registers is compared to the knot timer array in order to determine which path segment the path is currently in.

W 68 CTPATH Segment counter - **PATH_SEG** which determines which path segment the path is currently in.

I*8 69 Position offset array - an integer array which is used to offset the final calculation of commanded position.

W 70 High Memory - word register which determines how much program buffer has been set aside for reserved memory operations. **HIMEM**

W 71 CPATH number of knots - the number of knots that are in the last CPATH. **N_KNOTS**

W 72 CPATH Number of axes - the number of axes that are included in the last CPATH. **N_AXES**

B 73 Current path selection - Will be set to a value between 1 and the MAX$PATH value; that being the maximum number of paths permitted in the path table. **CURR$PATH**

W 74 Straight line number of knots - register which determines the maximum number of knots which will be used in the calculation of all straight line moves. **ST_PATH_KNOTS**

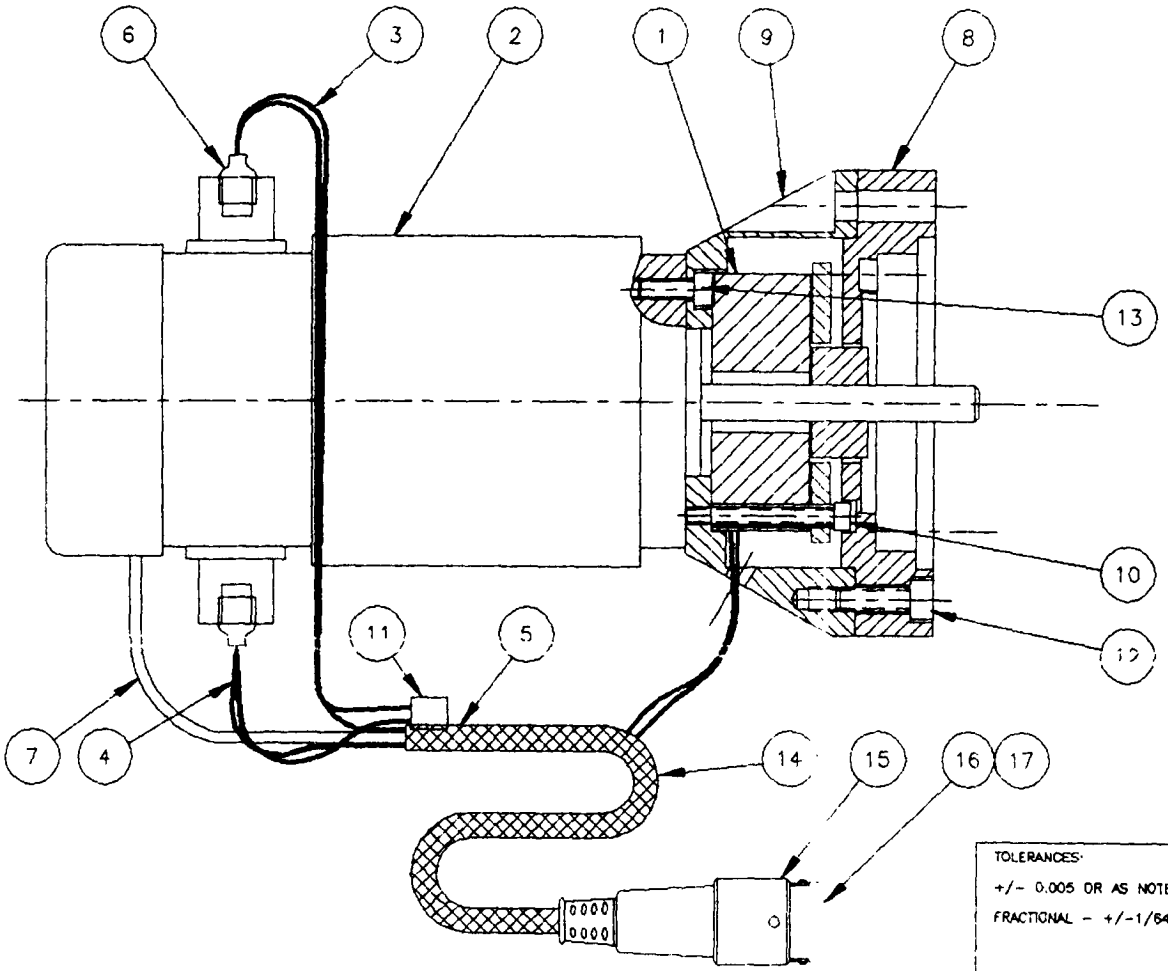
B 75 Teach counter maximum - byte value which determines how many teach locations will be permitted during this teach session.

B 76 Enable NULL positioning mode - In this mode, the finish criterion is not complete until each axis has moved to within a preset tolerance band of the end point. **NULL_ON**

B*8 77 Null positioning tolerance - the tolerance band for NULL positioning is set for each axis. Byte values allow for 127 pulse positioning error. **NULL_TOLERANCE**

B 78 Trigger enabled flag - Enable the trigger events during the next CPATH, CTPATH. **TRIGGER_ENABLE**

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
18	.17'	R-WHS/01BKXX HEAT SHRINK TUBE
17	3	R-CTR/INP-18 INSERT PINS
16	5	R-CTR/INP-24 INSERT PINS
15	1	R-CTR/12P-RC ROUND CONNECTOR PLUG
14	.75	R-WSV/04NYLN 1/4 GP NYLON BRAID
13	4	R-F0632SHC08 CAP SCREW
12	2	R-F0832SHC08 CAP SCREW
11	1	R-CAM103R050 CAPACITOR
10	3	R-F0440SHC16 CAP SCREW
9	1	T-ABK-M91416 BRAKE HOUSING JT 2/3
8	1	T-ABK-M91415 BRAKE H/D FLANGE
7	.50'	R-WHS/03BKXX HEAT SHRINK TUBING
6	2	R-WSL/03SPAD SPADE LUG
5	1	R-WCT/12D060 WIRE TIE
4	1.13'	R-WHU/18RDXX WIRE
3	1.13'	R-WHU/18BKXX WIRE
2	1	R-MTR/ML2110 DC SERVO MOTOR/ENCDR.
1	1	R-TRC/FSB003 FAIL-SAFE BRAKE

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A151/251 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 2/3 MOTOR ASS'Y

ASS'Y No: S-SMC-14-127

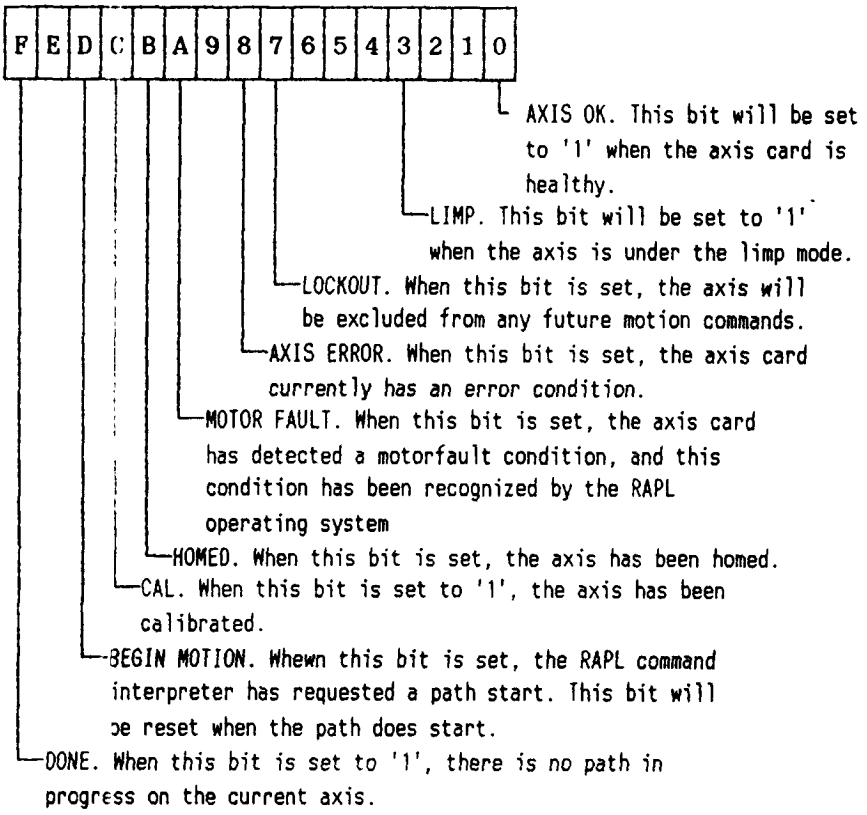
	DATE	NAME
DRAWN	DEC. 14, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SD-14-103

PART No: S-SMC-14-103

Data Type	Offset in List	Item Description (This pointer points to..)

W*8 91 Axis status array pointer - Array of 8 words which defines the status of the controller axes. This array is organized as a bit array. It replaces the old byte flag arrays: DONE, BEGIN_MOTION, LIMP, LOCKOUT. **AXIS_STATUS**. The bit allocation of this word array is shown below:



WARNING: All unused bits are reserved by the RAPL operating system. Also, modification of these registers is strictly not recommended, as they are constantly updated by the RAPL communication software with the smart axis cards. Use these registers as status indicators only.

Table G-1

Subfunction 03

Description

Return True/False result of a test for a receive character ready in the input character buffer. This is not an indication of the state of the hardware input buffer, but the RAPL-II character FIFO buffer.

Inputs

AH = 3

Outputs

AL = 00000000B - character not ready
AL = XXXXXXX1B - character ready

Subfunction 04

Description

Return True/False result of a test for an available spot in the transmit buffer. This is an indication of the state of the hardware transmit buffer, as RAPL-II issues characters directly to the output channel without buffering.

Inputs

AH = 4

Outputs

AL = 00000000B - buffer not ready
AL = XXXXXXX1B - buffer ready

Subfunction 05

Description

Transmit a NULL terminated string to the default output device.

Inputs

AH = 5
ES:[BX] points to a NULL terminated string

Outputs

None

Program Table

The program table is next. It can best be described by the following 'C' structure reference:

```
struct p_table
  {
  char name[8];
  int index;
  int checksum;
  } progtable[prog_table_size], *prog_table_ptr;
```

The parameter pointer PROG_TABLE_PTR points to the table in memory, while the word parameter PROG_TABLE_SIZE defines the number of entries in the table.

Each entry into the program table takes 12 bytes. The 'index' element defines the starting point in the program buffer for that particular program. The actual location of the program is then referenced by:

```
char *p; /* 'p' can point to the program in question */
int i;   /* 'i' is the index of the program in the program table */

    p = &(prog_buff_ptr[progtable[i].index]);
```

The checksum byte is the addition of all the program bytes up to the final '$' character. The checksum is only a byte length addition, so the upper byte of the checksum word is undefined.

Location Table

The location table is next. It can be described by the following 'C' structure reference:

```
struct l_table
  {
  char name[8];
  float locn[8];
  int checksum;
  } *loc_table_ptr, loctable[loc_table_size];
```

For a precision point, the location table would be described as:

```
struct l_table
  {
  char name[8];
  long locn[8];
  int checksum;
  } *loc_table_ptr, loctable[loc_table_size];
```

Thus any entry in the location table requires 42 bytes of data: an 8-byte name, 8 real or double-word fields (4 bytes each) and a checksum word. The checksum

APPENDIX I - RAPL-II BIOS INTERFACE

Introduction

The RAPL-II BIOS is an interface technique for those advanced users who wish to use 8086 machine code programs in conjunction with RAPL-II programs to provide a highly efficient robot control program. A 8086 machine code program can be written to support complex communication interfaces, or can be programmed to provide tracking control by creating offsets for the RAPL-II command interpolators. The extent to which this level of programming can affect the operation of the robot is quite extensive, so it is recommended that users consult with CRS Technical personnel to verify that proper operation will be maintained.

The RAPL-II BIOS interface is available with RAPL-II version 1.03 and later.

References

The following RAPL-II commands allow support for 8086 machine code programs:

HIMEM
EXECUTE
FREE
ALLOCATE

ROBCOMM-II version 1.0 and later supports downloading of machine code programs into the robot memory.

Use

8086 machine code programs can be executed from RAPL-II programs by using the EXECUTE command.

Description

The following is a description of the standard interface. The BIOS is accessible through software interrupt #69. The AX register contains the major and minor function codes that are described below. The AL register contains the major code, and the AH register contains the minor code. Additional machine code registers are used whenever needed, and return codes are described. Generally, the BIOS does not preserve either 8086 or 8087 registers, and the 8086 interrupts are immediately enabled once the BIOS is entered.

During execution of the path, RAPL scans two word values at each knot. Each contains a map of the lower 16 user output line numbers. A "1" in a bit position in the first word represents an output whose state changes at that knot. The second word contains the desired state of that output after the change. The change is made at I/O scan rates so that the state will change within approximately 40 milliseconds from the time the commanded position of the robot reaches the knot.

Path Table

The path table is used to save information regarding t\all of the continuous paths that have been taught to the robot. Information in the table will indicate where the raw path data is located, how it is formatted, and what locations were used to program the path. The path table can be considered as the following 'C' structure.

```
struct path_table
  {
  char control;
  char checksum;
  long start_locn[8];
  int index;
  int n_axes;
  int n_knots;
  char template[5];
  byte speed;
  byte start_index;
  byte end_index;
  } pathtable[pathtablesize];
```

The 'control' element assumes a value of 0 when no path is loaded, and '1' when there is a valid path. The 'checksum' byte is the summation of the raw path data.
The start_locn array is the precision point of the path start. Once a path is loaded, it no longer takes tool transforms into account, so a precision point must be saved as the starting location of the path. The robot will attempt a joint interpolated move to this location before the continuous path control takes over. The 'index' element points to the start of the path data in the path data buffer. 'n_axes' is the number of axes under which the path was taught, and 'n_knots' describes the number of path knots that were programmed. 'template' is the teach name template used for all of the path knots, and the 'start_index' and 'end_index' is used to describe which locations were used to teach the path. 'speed' was the programmed path speed.

APPENDIX H -- 'C' PROGRAMMING HINTS

This Appendix will provide a brief example of programming a host computer in the 'C' language to deal with internal RAPL parameters and user memory. Table G-1 provides a list of pointers to the majority of RAPL parameters. This information can be utilized very easily by a 'C' program, since 'C' can deal very well with pointers.

The following is a 'C' program section which would deal with this pointer list. It assumes that the programmer has developed an ACI interface package, or that the programmer has purchased the CRS ACI protocol software development package.

For the puroses of this example, review table G-1, with particular emphasis on the first 9 pointers.

In this example, let us assume that the pointer list has been read in, so that variable work_space_ptr now contains a valid pointer to the user space. In addition, the sizes of all of the user space elements have also been read in. We can now build the pointers to the various elements of user memory;

```
.
.
/*
This first argument is a pointer to the pointer list location in IBM memory, and
the second argument is a count of the number of pointers that are to be read from
the robot into memory.
*/
robotptrs( **char, int )
.

/* Pointer and structure definitions */
char *ptr_list[10]; /* Space for 10 pointers - that is all we need */
#define WS_PTR     ptr_list[0] /* pointer to user space pointer */
#define PT_PTR     ptr_list[3]/* pointer to program table pointer */
#define VT_PTR     ptr_list[5]/* pointer to variable table pointer */
#define LT_PTR     ptr_list[7]/* pointer to location table pointer */
#define LT_SIZ_PTR ptr_list[8]/* pointer to location table size parameter */
#define VT_SIZ_PTR ptr_list[6]/* pointer to variable table size parameter */
#define PT_SIZ_PTR ptr_list[4]/* pointer to program table size parameter */
#define PB_SIZ_PTR ptr_list[1]/* pointer to program buffer size parameter */
#define PB_CNT_PTR ptr_list[2]/* Pointer to program buffer count param */

char *prog_buff;
unsigned int pb_cnt, pb_size, lt_size, pt_size, vt_size;

struct prog_table
   {
   char prog_name[8];
   int prog_ptr;
   int checksum;
   } *prog_table_ptr;
```

These memory regions can be decribed as shown here:

.

.

float pknots[n_axes][n_knots]:

The knots array is located at the start of path memory. The knots array is actually a two dimensional array, bounded by the number of knots and the number of axes as the two dimensions. The total array size is therefore:

n1 = #knots * #axes * 4, in bytes.

float ptime[n_knots];

The time array is a float array that marks the elapsed path time at each knot.

float paccel[n_axes][n_knots];

The acceleration array contains real numbers marking the acceleration of each axis at each knot position.

float ptemp[n_knots];

long ptrig[n_knots].

This array is used for temporary storage during the calculation of the path parameters. It is then used to store the trigger setup information for the path. When storing the trigger information, the format of the data are two 16 bit word masks for each knot. The lower order word contains the bits which determine the outputs that are to be controlled at that particular knot. A 1 bit will indicate that the corresponding output will be controlled. The higher order word determines the state to which the designated outputs will be placed. A 0 bit will turn an output on, and a 1 will turn it off.

With this information, all of the pointers can be formulated, as the following code extraction demonstrates.

```
.
.
.
float *tp_pknots, *tp_paccel, tp_ptemp;
.
.
pknots = &pathbuffer[pathtable[pathnum].index]);
ptime = &(pknots[n_knots][n_axes]);
paccel = &(ptime[n_knots]);
ptrig = ptemp = &(paccel[n_knots][n_axes]);
.
.
```

These memory regions can be decribed as shown here:

```
.
.
float pknots[n_axes][n_knots]:
```

The knots array is located at the start of path memory. The knots array is actually a two dimensional array, bounded by the number of knots and the number of axes as the two dimensions. The total array size is therefore:

n1 = #knots * #axes * 4, in bytes.

```
float ptime[n_knots];
```

The time array is a float array that marks the elapsed path time at each knot.

```
float paccel[n_axes][n_knots];
```

The acceleration array contains real numbers marking the acceleration of each axis at each knot position.

```
float ptemp[n_knots];
long     ptrig[n_knots];
```

This array is used for temporary storage during the calculation of the path parameters. It is then used to store the trigger setup information for the path. When storing the trigger information, the format of the data are two 16 bit word masks for each knot. The lower order word contains the bits which determine the outputs that are to be controlled at that particular knot. A 1 bit will indicate that the corresponding output will be controlled. The higher order word determines the state to which the designated outputs will be placed. A 0 bit will turn an output on, and a 1 will turn it off.

With this information, all of the pointers can be formulated, as the following code extraction demonstrates.

```
.
.
.
float *tp_pknots, *tp_paccel, tp_ptemp;
.
.
pknots = &pathbuffer[pathtable[pathnum].index]);
ptime = &(pknots[n_knots][n_axes]);
paccel = &(ptime[n_knots]);
ptrig = ptemp = &(paccel[n_knots][n_axes]);
.
.
```

This Appendix will provide a brief example of programming a host computer in the 'C' language to deal with internal RAPL parameters and user memory. Table G-1 provides a list of pointers to the majority of RAPL parameters. This information can be utilized very easily by a 'C' program, since 'C' can deal very well with pointers.

The following is a 'C' program section which would deal with this pointer list. It assumes that the programmer has developed an ACI interface package, or that the programmer has purchased the CRS ACI protocol software development package.

For the puroses of this example, review table G-1, with particular emphasis on the first 9 pointers.

In this example, let us assume that the pointer list has been read in, so that variable work_space_ptr now contains a valid pointer to the user space. In addition, the sizes of all of the user space elements have also been read in. We can now build the pointers to the various elements of user memory;

```
.
.
/*
This first argument is a pointer to the pointer list location in IBM memory, and
the second argument is a count of the number of pointers that are to be read from
the robot into memory.
*/
robotptrs( **char, int )
.

/* Pointer and structure definitions */
char *ptr_list[10]; /* Space for 10 pointers - that is all we need */
#define WS_PTR     ptr_list[0] /* pointer to user space pointer */
#define PT_PTR     ptr_list[3]/* pointer to program table pointer */
#define VT_PTR     ptr_list[5]/* pointer to variable table pointer */
#define LT_PTR     ptr_list[7]/* pointer to location table pointer */
#define LT_SIZ_PTR ptr_list[8]/* pointer to location table size parameter */
#define VT_SIZ_PTR ptr_list[6]/* pointer to variable table size parameter */
#define PT_SIZ_PTR ptr_list[4]/* pointer to program table size parameter */
#define PB_SIZ_PTR ptr_list[1]/* pointer to program buffer size parameter */
#define PB_CNT_PTR ptr_list[2]/* Pointer to program buffer count param */

char *prog_buff;
unsigned int pb_cnt, pb_size, lt_size, pt_size, vt_size;

struct prog_table
   {
   char prog_name[8];
   int prog_ptr;
   int checksum;
   } *prog_table_ptr;
```

During execution of the path, RAPL scans two word values at each knot. Each contains a map of the lower 16 user output line numbers. A "1" in a bit position in the first word represents an output whose state changes at that knot. The second word contains the desired state of that output after the change. The change is made at I/O scan rates so that the state will change within approximately 40 milliseconds from the time the commanded position of the robot reaches the knot.

Path Table

The path table is used to save information regarding t\all of the continuous paths that have been taught to the robot. Information in the table will indicate where the raw path data is located, how it is formatted, and what locations were used to program the path. The path table can be considered as the following 'C' structure.

```
struct path_table
  {
  char control;
  char checksum;
  long start_locn[8];
  int index;
  int n_axes;
  int n_knots;
  char template[5];
  byte speed;
  byte start_index;
  byte end_index;
  } pathtable[pathtablesize];
```

The 'control' element assumes a value of 0 when no path is loaded, and '1' when there is a valid path. The 'checksum' byte is the summation of the raw path data.
The start_locn array is the precision point of the path start. Once a path is loaded, it no longer takes tool transforms into account, so a precision point must be saved as the starting location of the path. The robot will attempt a joint interpolated move to this location before the continuous path control takes over. The 'index' element points to the start of the path data in the path data buffer. 'n_axes' is the number of axes under which the path was taught, and 'n_knots' describes the number of path knots that were programmed. 'template' is the teach name template used for all of the path knots, and the 'start_index' and 'end_index' is used to describe which locations were used to teach the path. 'speed' was the programmed path speed.

APPENDIX I - RAPL-II BIOS INTERFACE

Introduction

The RAPL-II BIOS is an interface technique for those advanced users who wish to use 8086 machine code programs in conjunction with RAPL-II programs to provide a highly efficient robot control program. A 8086 machine code program can be written to support complex communication interfaces, or can be programmed to provide tracking control by creating offsets for the RAPL-II command interpolators. The extent to which this level of programming can affect the operation of the robot is quite extensive, so it is recommended that users consult with CRS Technical personnel to verify that proper operation will be maintained.

The RAPL-II BIOS interface is available with RAPL-II version 1.03 and later.

References

The following RAPL-II commands allow support for 8086 machine code programs:

HIMEM
EXECUTE
FREE
ALLOCATE

ROBCOMM-II version 1.0 and later supports downloading of machine code programs into the robot memory.

Use

8086 machine code programs can be executed from RAPL-II programs by using the EXECUTE command.

Description

The following is a description of the standard interface. The BIOS is accessible through software interrupt #69. The AX register contains the major and minor function codes that are described below. The AL register contains the major code, and the AH register contains the minor code. Additional machine code registers are used whenever needed, and return codes are described. Generally, the BIOS does not preserve either 8086 or 8087 registers, and the 8086 interrupts are immediately enabled once the BIOS is entered.

Program Table

The program table is next. It can best be described by the following 'C' structure reference:

```
struct p_table
  {
  char name[8];
  int index;
  int checksum;
  } progtable[prog_table_size], *prog_table_ptr;
```

The parameter pointer PROG_TABLE_PTR points to the table in memory, while the word parameter PROG_TABLE_SIZE defines the number of entries in the table.

Each entry into the program table takes 12 bytes. The 'index' element defines the starting point in the program buffer for that particular program. The actual location of the program is then referenced by:

```
char *p; /* 'p' can point to the program in question */
int i;   /* 'i' is the index of the program in the program table */

    p = &(prog_buff_ptr[progtable[i].index]);
```

The checksum byte is the addition of all the program bytes up to the final '$' character. The checksum is only a byte length addition, so the upper byte of the checksum word is undefined.

Location Table

The location table is next. It can be described by the following 'C' structure reference:

```
struct l_table
  {
  char name[8];
  float locn[8];
  int checksum;
  } *loc_table_ptr, loctable[loc_table_size];
```

For a precision point, the location table would be described as:

```
struct l_table
  {
  char name[8];
  long locn[8];
  int checksum;
  } *loc_table_ptr, loctable[loc_table_size];
```

Thus any entry in the location table requires 42 bytes of data: an 8-byte name, 8 real or double-word fields (4 bytes each) and a checksum word. The checksum

Subfunction 03

Description

Return True/False result of a test for a receive character ready in the input character buffer. This is not an indication of the state of the hardware input buffer, but the RAPL-II character FIFO buffer.

Inputs

AH = 3

Outputs

AL = 00000000B - character not ready
AL = XXXXXXX1B - character ready

Subfunction 04

Description

Return True/False result of a test for an available spot in the transmit buffer. This is an indication of the state of the hardware transmit buffer, as RAPL-II issues characters directly to the output channel without buffering.

Inputs

AH = 4

Outputs

AL = 00000000B - buffer not ready
AL = XXXXXXX1B - buffer ready

Subfunction 05

Description

Transmit a NULL terminated string to the default output device.

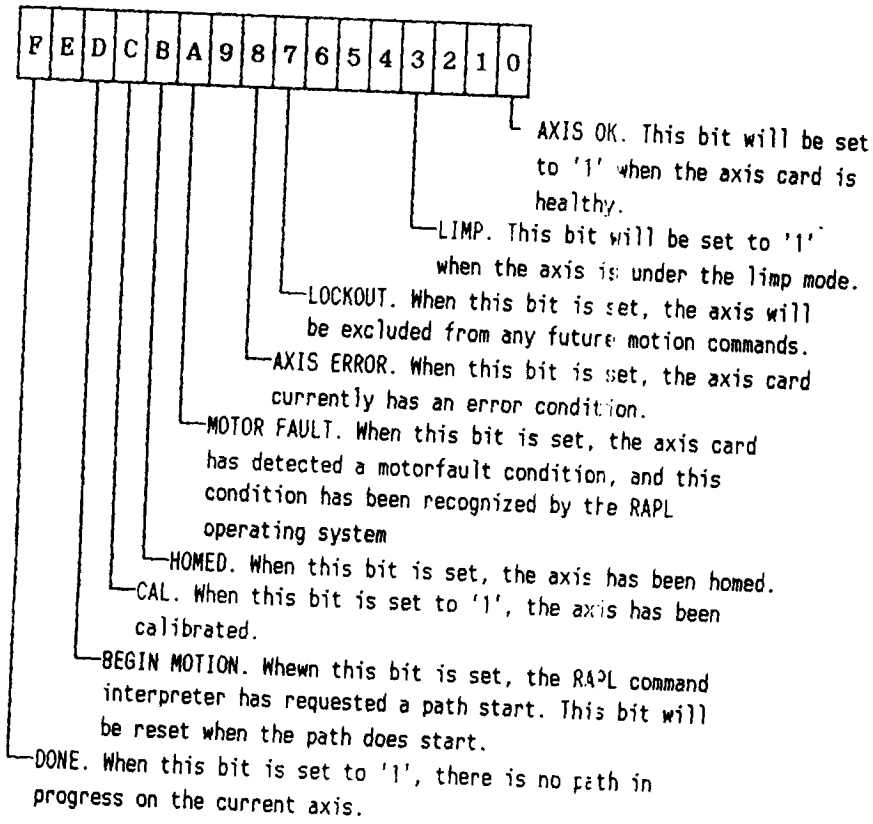
Inputs

AH = 5
ES:[BX] points to a NULL terminated string

Outputs

None

Data Type	Offset in List	Item Description (This pointer points to..)
W*8	91	Axis status array pointer - Array of 8 words which defines the status of the controller axes. This array is organized as a bit array. It replaces the old byte flag arrays: DONE, BEGIN_MOTION, LIMP, LOCKOUT. **AXIS_STATUS**. The bit allocation of this word array is shown below:



WARNING: All unused bits are reserved by the RAPL operating system. Also, modification of these registers is strictly not recommended, as they are constantly updated by the RAPL communication software with the smart axis cards. Use these registers as status indicators only.

Table G-1

Function 03

Description
Name string handling routines. RAPL-II deals with names as special strings, so it is useful to have conversion facilities for those programmers who deal with NULL strings, or other string types.

Common Inputs
AL = 03

Subfunction 01
Description
Pad string with spaces to create RAPL-II standard table look-up name. This array can be NULL terminated. The array must be able to hold up to 8 characters. The name will be padded with spaces to provide compatibility with RAPL name matching functions

Inputs
AH = 1
ES:[BX] points to a name string.

Outputs
ES:[BX] points to the padded name string.

Subfunction 02
Description
Pad a name string with '_' characters to create a RAPL-II standard teach template.

Inputs
AH = 2
ES:[BX] points to a name string.

Outputs
ES:[BX] points to the padded name array.

Data Type	Offset in List	Item Description (This pointer points to..)

W 66 Size of the CTPATH buffer - Pointer to the word parameter which contains the number of bytes allocated for the path buffer. **PATH_BUFFER_SIZE**

R 67 CTPATH Timer - **TIMER** which increments as the continuous path proceeds. This timer registers is compared to the knot timer array in order to determine which path segment the path is currently in.

W 68 CTPATH Segment counter - **PATH_SEG** which determines which path segment the path is currently in.

I*8 69 Position offset array - an integer array which is used to offset the final calculation of commanded position.

W 70 High Memory - word register which determines how much program buffer has been set aside for reserved memory operations. **HIMEM**

W 71 CPATH number of knots - the number of knots that are in the last CPATH. **N_KNOTS**

W 72 CPATH Number of axes - the number of axes that are included in the last CPATH. **N_AXES**

B 73 Current path selection - Will be set to a value between 1 and the MAX$PATH value; that being the maximum number of paths permitted in the path table. **CURR$PATH**

W 74 Straight line number of knots - register which determines the maximum number of knots which will be used in the calculation of all straight line moves. **ST_PATH_KNOTS**

B 75 Teach counter maximum - byte value which determines how many teach locations will be permitted during this teach session.

B 76 Enable NULL positioning mode - In this mode, the finish criterion is not complete until each axis has moved to within a preset tolerance band of the end point. **NULL_ON**

B*8 77 Null positioning tolerance - the tolerance band for NULL positioning is set for each axis. Byte values allow for 127 pulse positioning error. **NULL_TOLERANCE**

B 78 Trigger enabled flag - Enable the trigger events during the next CPATH, CTPATH. **TRIGGER_ENABLE**

Subfunction 03

Description

Return the value from the RAPL-II variable table into the local structure. If the variable did not previously exist, then it will be created and a value of 0 will be returned.

Inputs

AH = 3

Outputs

Structure value will be updated.

Function 05

Description

Access to the RAPL-II location table is provided through this function level. Data can be removed from, or added to the RAPL-II location table. Often, it is more efficient to pull data from the table and use it and then return the data.

Common Inputs

AL = 05

ES:[BX] points to a location table structure, either as a source for data, or as a destinatior for data extracted from the table. A location table structure is defined by the following 'C' reference:

```
struct cartesian_loctable
   {
   char name[8];
   float cartesian[8];
   int checksum;
   }
struct precision_loctable
   {
   char name[8];
   long point[8];
   int checksum;
   }
```

In no case is the checksum used by the BIOS; this structure is only used to maintain compatibility with RAPL-II.

Data Type	Offset in List	Item Description (This pointer points to..)

B 42 Teach name count - The byte value which is encoded into the final three character positions of the teach name when the point is finally stored. **TEACH_NAME_COUNT**

B 43 Program pause flag - When a program is executing, setting this byte to a '1' value will stop the program after the next line is decoded. **PROGRAM_PAUSE**

B 44 Homing complete flag - This flag is set when the robot has been homed after a power up. Do not confuse this with item 44, which indicates whether or not the robot has been calibrated. This flag must be set in order that all move functions that access points will work. **ROBOT_IS_HOMED**

B*128 45 String buffer - The string buffer is a buffer 128 bytes long, that is divided evenly into 4 32 byte areas. These correspond to the 4 string variables that are allowed by RAPL version 3.50 or later. **STRING_BUFFER**

B 46 Loss of feedback/collision detection byte - This byte indicates the status of this function. A boolean true means that the function is turned on. **LOFB_CHECK**

B 47 Program completed flag - indicates whether or not a program is completed. **PROGRAM_COMPLETED**

W 48 Program repetitions counter - word counter that indicates the number of repetitions that the current program has completed. **PROGRAM_REPETITIONS**

W 49 Program repetitions requested - the word register which indicates the number of repetitions of the program that have been requested. **PROGRAM_REPEAT_LIMIT**

B 50 Loop forever Flag - the flag which indicates that an infinite looping of the current program has been requested. **PROGRAM_LOOP_FOREVER**

R*250 51 Straight-line Buffer - the 1000 byte buffer which contains the data for the current straight line move. Also used to contain the path parameters for a CPATH. **STRAIGHT_LINE_BUFFER**

B 52 Trigger Enabled flag - flag which indiactes whether or not the trigger table has been enabled. **TRIGGER_TABLE**

Function 07

Description

Access to the RAPL-II transformation equations. Depending upon the current RAPL-II configuration, transformations can be utilized. The 8087 is typically used for these equations, so if this code is executing from within a RAPL-II interrupt, the 8087 status should be maintained properly. All transformation arrays should provide for 32 bytes for input and output. (8 reals or 8 longs)

Inputs

ES:[BX] points to the input array.

AH = 1 motor to joint transformation. Input array in motor pulses, output array in joint values.

AH = 2 joint to motor transformation. Inverse of function 1.

AH = 3 joint to world transformation. Input in joint values, output in world coordinate system.

AH = 4 world to joint transformation. Inverse of function 3.

AH = 5 motor to world transformation. Input in motor pulses. Output in world coordinates.

AH = 6 world to motor transformation. Inverse of function 5.

Outputs

DX:[DI] points to the output array

Function 08

Description

RAPL-II BIOS test code. To test the user's interface code, the BIOS will return a known result. This is for software development only.

Inputs

AL = 08

Outputs

AX Test return value 0A5A5H

Data Type	Offset in List	Item Description (This pointer points to..)

R*8 24 Acceleration array - An array of 8 reals which determine the acceleration of each motor during a joint interpolated motion. **ACCELERATION(8)**

R*8 25 Transmission Ratio array - An array of 8 real numbers which equates the measurement units of the joint or motor with the corresponding motor pulse resolution. It is dangerous to change the first five values in this array, as it will change the robot transformation. The @XRATIO command changes the other three.

B 26 Alarm Number - A byte value that contains the last RAPL error code to be generated. **ALARM_NUMBER**

W*8 27 Rotary Resolution array - An array of 8 integers which specifies the motor encoder resolution in encoder lines per motor revolution. **ROTARY_RESOLUTION(8)**

R*6 28 Tool Transform - an array of 6 real numbers that specify the current tool transform value. **TOOL_TRANSFORM**

Struct 29 Input block structure - the RAPL character input buffer area. The input buffers are specified in the following format:

```
input_block(3) structure (
     buffsize byte,
in_ptr byte,
out_ptr byte,
buffer(128) byte)
```

The first two structure records keep track of data input from the two serial ports. The third is used as an intermediate buffer when executing from a program. 'buffsize' determines the size of the input ring buffer, up to 128 bytes. 'in_ptr' points to the next array element available for serial input, and 'out_ptr' marks the next character that can be released from the buffer, to be used by RAPL.

B 30 Alarm status - A boolean byte value which specifies whether or not a RAPL error has occurred. The byte is reset when a new command is attempted. **ALARM_STATUS**

R*8 31 Motor to Joint transform factor array - A real array which transforms the motor coordinates into the joint coordinates.

32 **NULL**

Function 0B

Description
Perform digital input/output functions.

Common Inputs
AL = 0B

Subfunction 01
Description
Output to digital channels. This interface processes I/O through the RAPL-II digital I/O buffers which are updated at 0.040 second intervals.

Inputs
AH = 1
CL = output value of 0 or 1
CH = channel number (0-71)

Outputs
None

Subfunction 02
Description
Read digital input channel.

Inputs
AH = 2
CH = channel number (0-71)

Outputs
AL = digital input value of 0 or 1.

Function 0C

Description
Process a RAPL-II DELAY function.

Inputs
AL = 0C
CX = delay count, in milliseconds.

Outputs

Data Type	Offset in List	Item Description (This pointer points to..)
W	4	Program table size - the Word register which tells us how long the program table is in number of total program entry locations. **PROG_TABLE_SIZE**
P	5	Variable table pointer - the Pointer to the variable table. **VAR_TABLE_PTR**
W	6	Variable table size - Word register which tells us how big the variable table is in the total number of entries. **VAR_TABLE_SIZE**
P	7	Location table pointer - Pointer to the location table. **LOC_TABLE_PTR**
W	8	Location table size - Word register which tells us how long the location table is in the total number of table entries. **LOC_TABLE_SIZE**
DI*8	9	Calibration register pointer - Pointer to an array of 8 double integers which contain the robot calibration position. **CALIBRATION(8)**
B	10	Calibrate position checksum byte - The byte addition of all valid calibration data for the 5 robot axes only. **CALIBRATE_CHECKSUM**
B	11	'robot is calibrated' - check byte which is set after a calibration procedure has been run. **ROBOT_IS_CALIBRATED**
B	12	TRACK$SPEC register - Register contains the state of the extra axes, and whether they are included in the cartesian Xfrm or not.
-	13	**NULL**
DI*8	14	Actual Position array - An array of 8 double integers which are the absolute position registers of the robot feedback sub-system. Every 4 milliseconds, these values are updated with the incremental positions determined by the motor encoders. **POSITION**
DI*8	15	Position Command array - An array of 8 double integers which represent the absolute position command to the robot motors. Any path generation algorithm can create these values, and the servo loop function will compare these registers to the actual position registers and will create the appropriate commands. **POSITION_COMMAND**

Function
0E

Description
IAXIS communication support. The communication protocol with the axis cards of the M2 controller is supported here. Do not attempt to use this level without prior consultation with CRS Plus.

Common Inputs
AL = 0E
DL = Axis card number (0-7)

Subfunction 01
Description
Write a data word to the axis card

Inputs
AH = 1
CX = data word

Outputs
AX = 00000000 failed attempt
AX = XXXXXXX1 passed attempt

Subfunction 02
Description
Read a single data word from the axis card

Inputs
AH = 2
Outputs
CX = data word
AX = 00000000 failed attempt - no valid data
AX = XXXXXXX1 passed attempt - valid data

APPENDIX G - RAPL-2 CONTROL PARAMETER LIST

TABLE OF CONTENTS

APPENDIX J - THE USE OF EXTRA AXES

CONTENTS:

FIGURES:

APPENDIX F - CONTROL SYSTEM TECHNICAL DESCRIPTION

[This APPENDIX contains detailed descriptions of the operation and archi-tecture of the SRS-M1A Controller. It is intended for those who require specialized interfacing or plan operating software modifications. For access to this document, please contact your distributor or CRS Plus.]

J-2 USE OF EXTRA AXIS #6 WITH A TRACK

Hardware Considerations:

If axis 6 is a TRACK using MCS-M2110 or similar motor, the built in DC Amplifier can be used (output specifications are found in section J-5). To use the sixth channel amplifier an axis card either P-Type or PID-Type (A150 or A250) must be installed in slot number 6.

The signals to control the 6th motor can be split out from the robot arm cables using a track cable from CRS. Figure J-1 shows the typical layout of these cables. The length of the track cable must be determined by the installation details. The main length of the cable is flexible enough to be used reliably in a track cable-carrier. The parts required for the TRACK installation are:

Quantity	Description	Part Number
1	M2110 - Motor/Encoder set.	SEC-09-029
1	M2110CXX - Motor Cable (XX is 10, 25, or 40 Foot)	SCC-09-024/026/044
1	TRCK/CXX - Robot Track Cable (XX is 15, 20, or 25 foot)	SCC-14-528/529/530

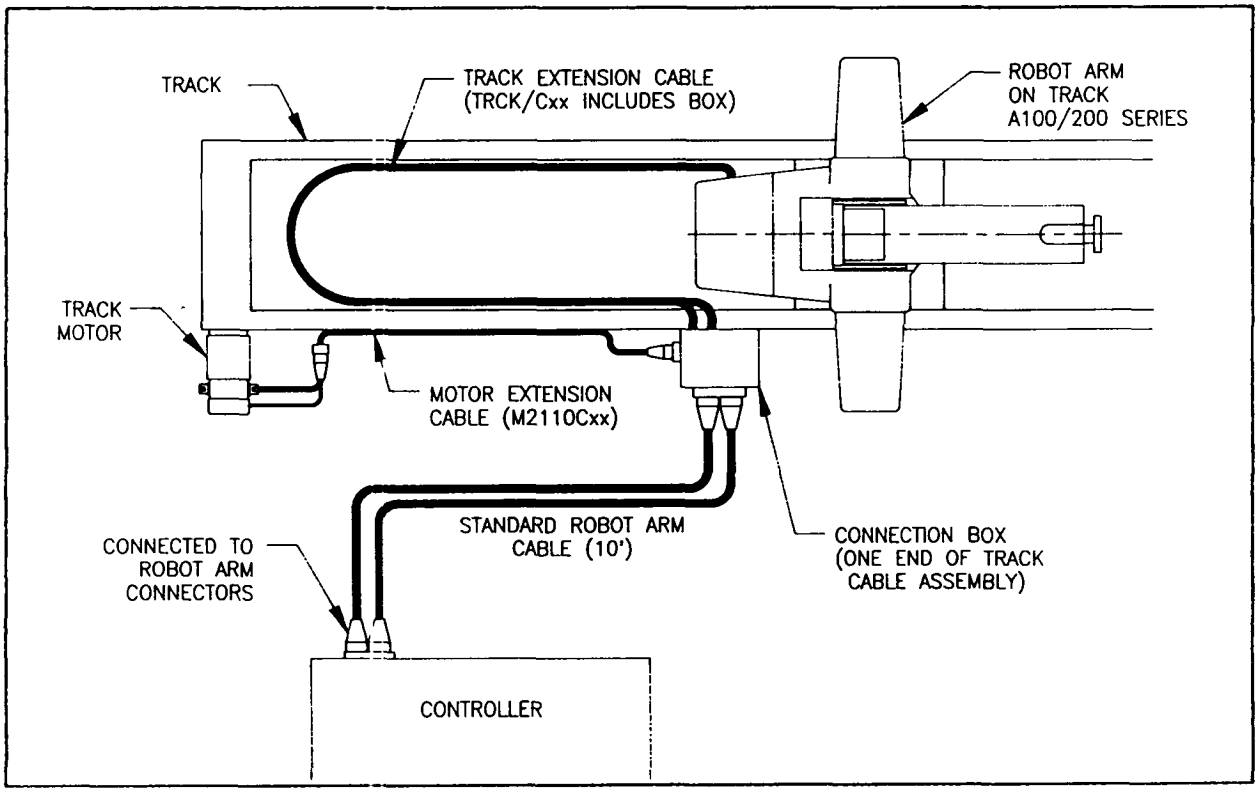


FIGURE J-1 - Typical track cable installation layout

APPENDIX E - OPTICAL ENCODER SPECIFICATIONS

The A100/200 controller can be interfaced to optical encoders with the following specifications:

- 2-channel (A, B) output with zero-cross signal or 4-channel (A, A*, B, B*) with zero-cross (Z, Z*).
- Square wave output.
- 5 volt DC power supply input with <120 milliamp current typ.
- Signal rise time of 1.0 microsecond max.
- Signal fall of 1.0 microsecond max.
- Pulse frequency range of DC to 50 kHz max.

J-2 USE OF EXTRA AXIS #6 WITH A TRACK (Continued)

5) Set up the maximum motor speed. Different amplifier/motor/load combinations produce a different maximum RPM. The controller must know the maximum RPM the joint can run at so it can command a reasonable speed. The number entered will be the rate of pulses commanded at a programmed speed of 100%.

```
>>@XMAXVEL AXIS #: 6, MAX JOINT VEL(RPM) = .286800000E+00004/ssss<cr>
```

Where **ssss** is the motor top speed in RPM.

6) Set the system up for TRACK operation. This means that all locations stored with the TRACK enabled will include the track position.

```
>>@TRACK "X" OR "Y" AXIS OR RESET: DIRECTION<cr>
```

Where **DIRECTION** is the direction of motion of the robot on the track relative to it's coordinate system. i.e. if the track moves the robot in the X direction then it is an X TRACK.

The above commands may be put into an initialization subroutine and get executed on each power up. RAPL-II allows the use of the PASSWORD within a program. A typical program might look like this:

```
PROGRAM TRACKSET:
1000 PASSWORD xxx
1100 ; The user would insert the correct password instead of xxx.
1200 @XPULSES 6,1000
1300 @XRATIO 6,2.5
1400 @XLIMITS 6,36,-36
1500 @XMAXVEL 6,1800
1600 @TRACK X
1700 PASSWORD WRONG
1800 RETURN
$
```

Track Calibration and Homing:

The following procedure is used to calibrate the track:

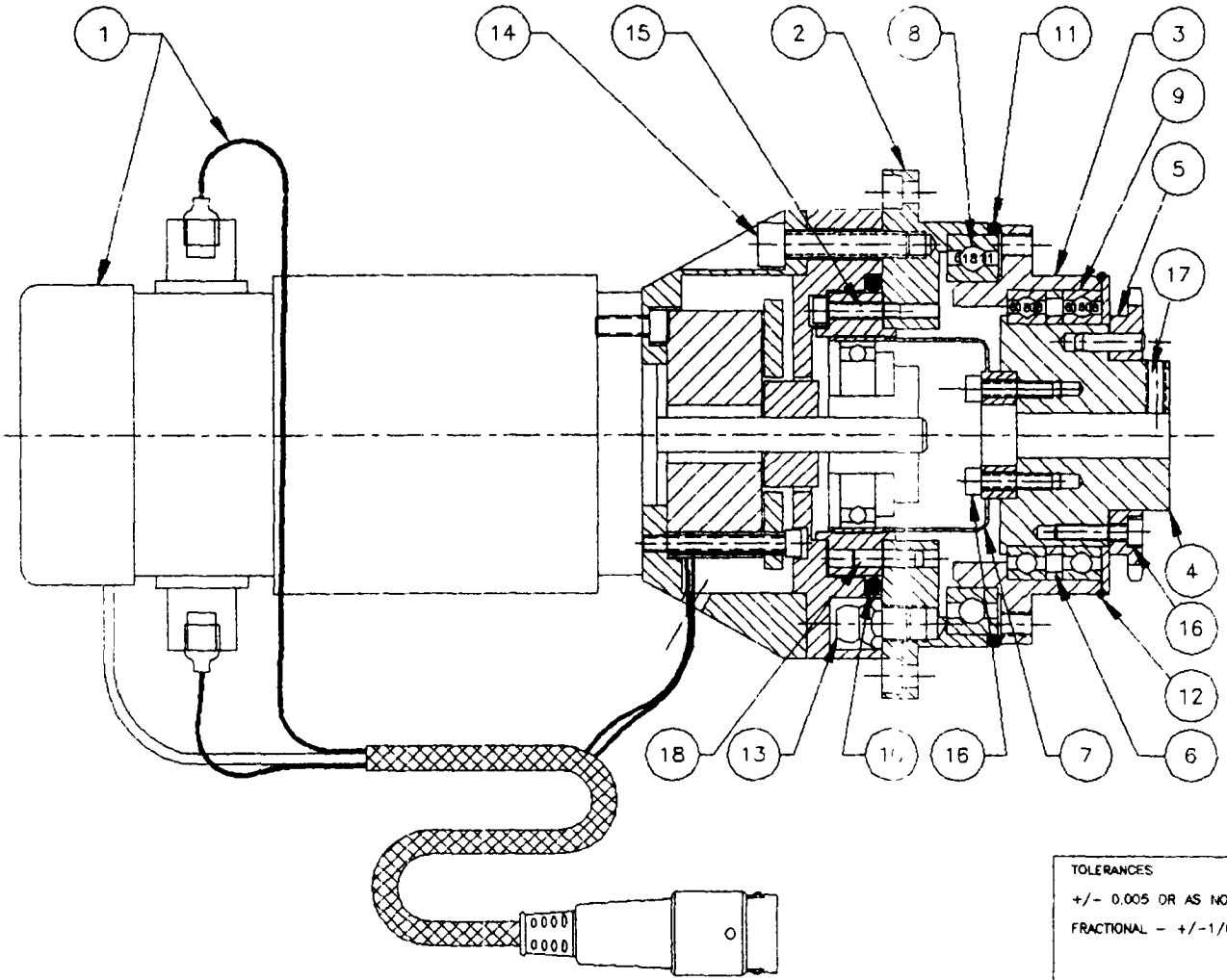
1) Turn on the MAIN AC power

2) Turn on the ARM power

3) Enter the PASSWORD

4) Put the robot in a MANUAL mode and use the switch labelled AXIS 6 to move the track. Be careful at this time not to collide with the ends of travel as the limits are not checked.

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
18	4	R-F02DLPIN08 DOWEL PIN
17	2	R-F0632CHS08 SET SCREW
16	10	R-F0440SHC08 CAP SCREW
15	6	R-F0440SHC10 CAP SCREW
14	4	R-F1024SHC14 CAP SCREW
13	1	R-GRF/LI5010 GREASE FITTING
12	1	R-GRO-RNG2.1 O-RING SEAL
11	1	R-GRO-RNG2.5 O-RING SEAL
10	1	R-GRO-RNG2ID O-RING SEAL
9	2	R-BG00061608 BEARING
8	1	R-BG00061811 BEARING
7	1	R-TRC/1C072A HARMONIC DRIVE
6	1	R-M/C-M91407 J3 BEARING SPACERS
5	1	R-M/C-M91410 JT 3 DRIVE SPROCKET
4	1	R-M/C-M91405 JOINT 3 PIVOT
3	1	T-ABK-M91404 J3 FLANGE
2	1	T-ABK-M91402 SHOULDER FLANGE
1	1	S-SMC-14-103 JOINT 2/3 MOTOR ASS'Y

PART LIST

TOLERANCES
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 163 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A151/251 ROBOT ARM

JOB No: CRS-914

TITLE: JOINT 3 DRIVE ASS'Y

ASS'Y No: S-RSA-14-103

	DATE	NAME
DRAWN	DEC. 15, 1989	P. D. Young
CHECKED		
SCALE	FULL SIZE	

DWG No: SB-14-129

PART No: S-SMC-14-129

Where nnn is the number of encoder counts per turn (used in the @XPULSES command). Place another mark on the track opposite the pointer now. The difference between the two marks is one full turn of the encoder. Both marks indicates the location of a marker pulse on the encoder. Filling in the space between the two marks shows the range within which the pointer can point before issuing the XHOME 6 command to reference the track.

10) Starting the system would involve moving the track to the middle of the homing mark and issuing the XHOME 6 command.

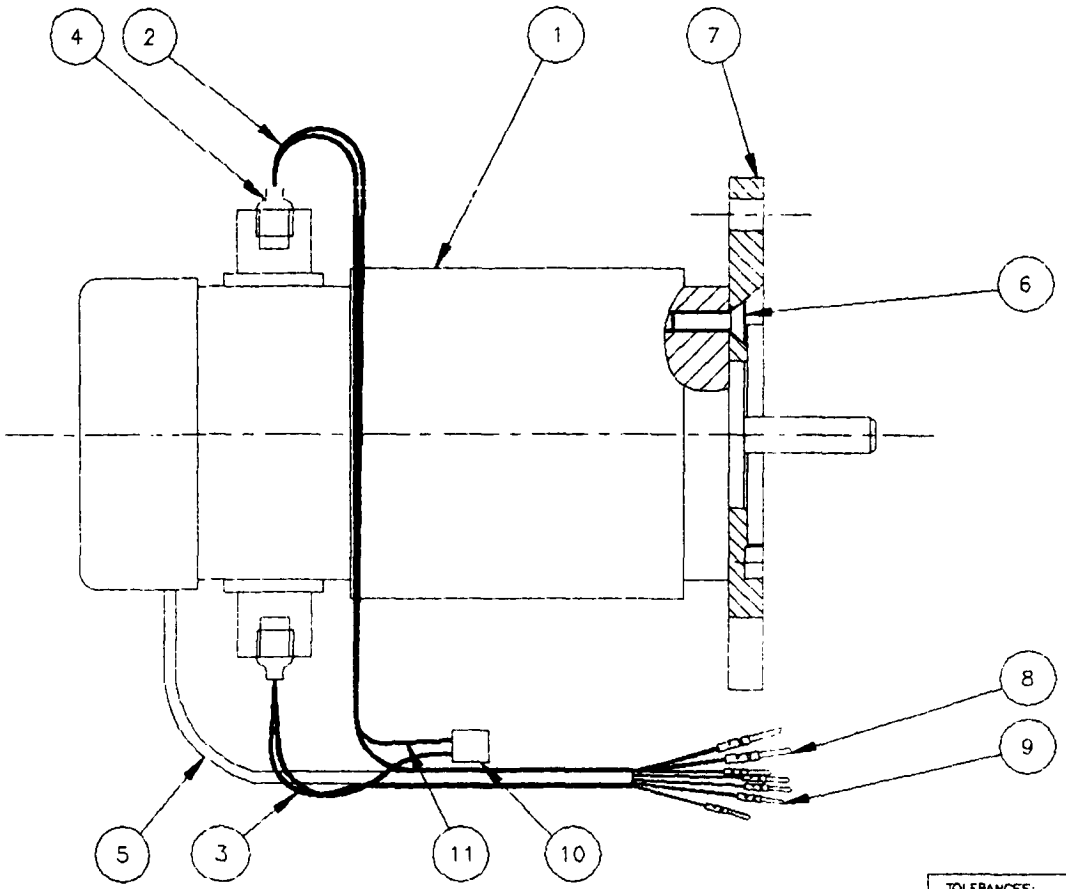
11) At any time issuing the command:

    >>XREADY AXIS #: 6<cr>

    will send the track to the location at which you zeroed it in step 6).

Your robot and track system is now ready for operation. Note that the READY and HOME commands are for the Robot arm only, you must use XHOME and XREADY commands for the track.

REVISION TABLE			
ITEM	DATE	DESCRIPTION	REL.



ITEM	REQ'D	PART NUMBER
11	.2'	R-WHS/01BKXX HEAT SHRINK TUBING
10	1	R-CAM103R050 CAPACITOR
9	5	R-CPC/CCS505 AMP 24 GA SOCKET INS.
8	2	R-CPC/CSS101 AMP 18 GA SOCKET INS.
7	1	T-ABK-M09224 H/D FLANGE
6	4	R-F0632FHC08 FLAT-HEAD SCREWS
5	.33'	R-WHS/03BKXX HEAT SHRINK TUBING
4	2	R-WSL/03SPAD SPADE LUG
3	.8'	R-WHU/18RDXX WIRE
2	.8'	R-WHU/18BKXX WIRE
1	1	R-MTR/ME2110 DC SERVO MOTOR/ENCDR.

PART LIST

TOLERANCES:
+/- 0.005 OR AS NOTED
FRACTIONAL - +/-1/64

MAT'L:

REQ'D:
1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

830 Harrington Court
P.O. Box 183 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM

JOB No: CRS-914

TITLE: MOTOR 1 ASSEMBLY

ASS'Y No: S-SMC-14-106

	DATE	NAME
DRAWN	DEC. 19, 1989	P. D. Young
CHECKED		

DWG No: SB-14-107

PART No: S-SMC-14-107

SCALE: FULL SIZE

J-3 USING EXTRA AXES #6 AND 7 AS A GANTRY (Continued)

4) Enter the joint limits.

```
>>@XLIMITS AXIS #: 6, POSITIVE JOINT LIMIT: +LIM6, NEGATIVE JOINT LIMIT: -LIM6<cr>
>>@XLIMITS AXIS #: 7, POSITIVE JOINT LIMIT: +LIM7, NEGATIVE JOINT LIMIT: -LIM7<cr>
```

5) Set up the maximum motor speed.

```
>>@XMAXVEL AXIS #: 6, MAX JOINT VEL(RPM) = .286800000E+00004/ssss<cr>
>>@XMAXVEL AXIS #: 7, MAX JOINT VEL(RPM) = .286800000E+00004/ssss<cr>
```

6) Set the system up for GANTRY operation. This means that all locations stored with the GANTRY enabled will include the GANTRY X and Y positions.

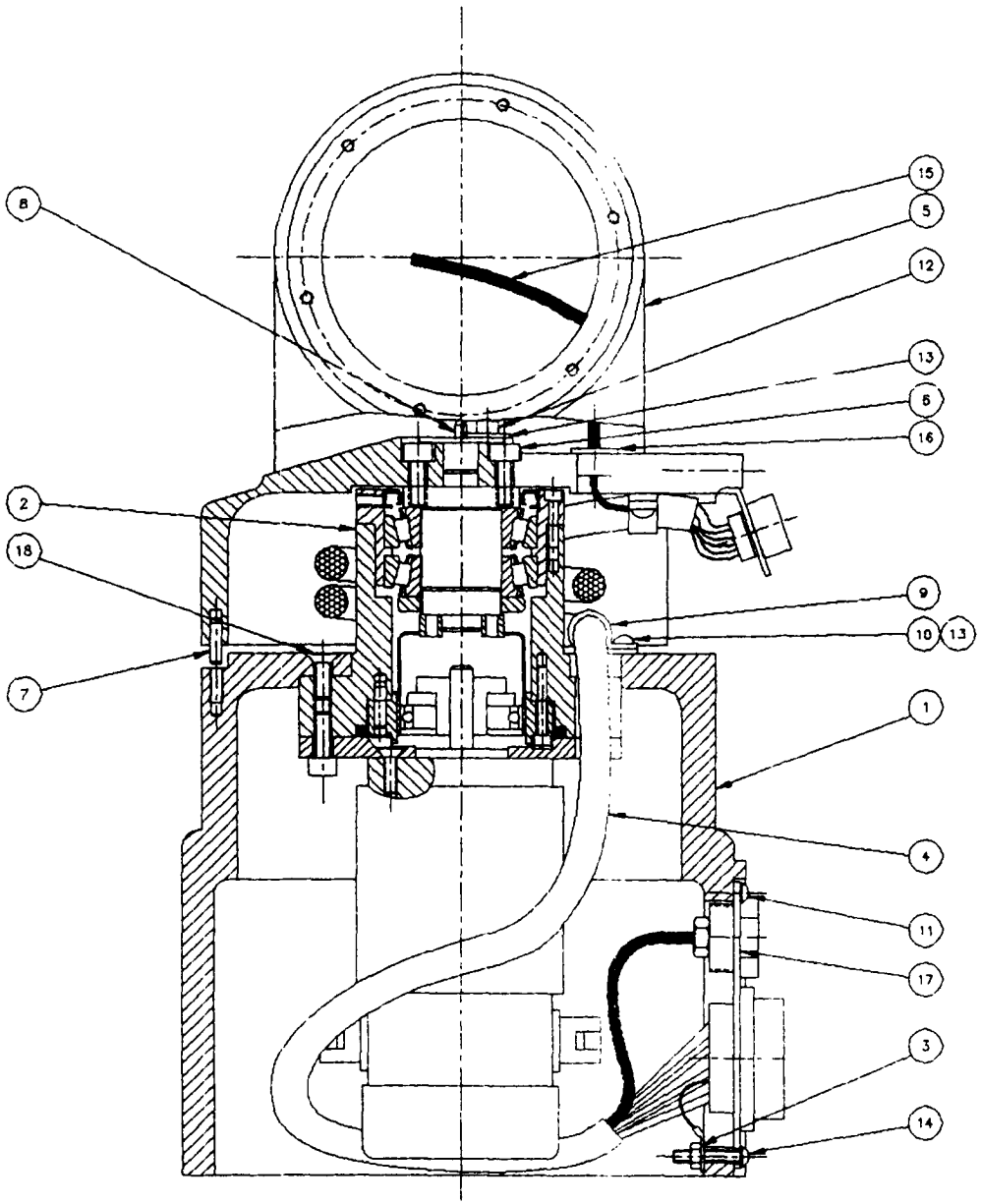
```
>>@GANTRY <cr>
```

This sets up axis 6 as the X axis of the GANTRY and axis 7 as the Y axis. Installation of the robot arm should correspond with this arrangement but this is not absolutely necessary.

The above commands may be put into an initialization subroutine and get executed on each power up. RAPL-II allows the use of the PASSWORD within a program. A typical program might look like this:

```
PROGRAM GNTRYSET:
1000 PASSWORD xxx
1100 ; The user would insert the correct password instead of xxx.
1200 @XPULSES 6,1000
1220 @XPULSES 7,1000
1300 @XRATIO 6,2.5
1320 @XRATIO 7,2.5
1400 @XLIMITS 6,36,-36
1420 @XLIMITS 7,72,-72
1500 @XMAXVEL 6,1800
1520 @XMAXVEL 7,1800
1600 @GANTRY
1700 PASSWORD WRONG
1800 RETURN
$
```

REVISION TA			
ITEM	DATE	DESCR... ION	REL.



ITEM	REQ'D	PART NUMBER
18	4	R-F1024FHC08 FLAT HEAD CAP SCREWS
17	1	R-ZSN/M1A-RA ARM S/N PLATE
16	1	R-WCC/06BU04 WIRE BUSHING 3/8ø
15	33	R-WHS/03BKXX HEAT SHRINK TUBING
14	1	R-F0632RRM12 MACHINE SCREW
13	4	R-F08FLTWASH #6 FLAT WASHER
12	2	R-F0632SHCS04 CAP SCREW
11	5	R-F0632RRM04 MACHINE SCREW
10	2	R-F0632RRM08 MACHINE SCREW
9	2	R-WCC/08IDD8 CABLE CLAMP
8	1	R-F02RLPIN16 ROLL PIN
7	2	R-F02DLPIN08 DOWEL PIN
6	4	R-F1024SHC06 CAP SCREWS
5	1	T-PBL-M91401 WAIST PAINTED BLACK
4	1	S-SMC-14-043 MAIN HARNESS ASSEMBLY
3	1	R-F0632HEXNT 6-32 HEX NUT
2	1	S-SMC-14-106 NECK ASSEMBLY
1	1	T-PBL-M91411 BASE PAINTED BLACK

PART LIST

TOLERANCES: +/- 0.005 OR AS NOTED
FRACTIONAL = +/-1/64

MAT'L:

REQ'D: 1 PER ASS'Y

FINISH:

CRSPLUS
INDUSTRIAL AUTOMATION

320 Harrington Court
P.O. Box 453 Station A
Burlington, Ontario
L7N 3N4

PROJECT: A150/250 ROBOT ARM			JOB No: CRS-914
TITLE: BASE SUBASSEMBLY			ASSY No: S-RSA-14-003
	DATE	NAME	DWG No: SC-14-104
DRAWN	DEC. 18, 1989	P. D. Young	
CHECKED			PART No: S-SMC-14-104

Where nnn is the number of encoder counts per turn for each axis (as used in the @XPULSES commands). Place another mark on the each track opposite the pointer now. The difference between the two marks on each track is one full turn of the encoder. Both marks indicates the location of a marker pulse on the encoder. Filling in the space between the two marks shows the range within which the pointer can point before issuing the XHOME command to reference each axis.

9) Starting the system will require moving the tracks to the middle of each homing mark and issuing the following commands:

```
>>XHOME AXIS #: 6<cr>
LOCATED IN HOME BOUNDS? Y<CR>
WAITING FOR HOME AXIS #006 **PASSED
>>XHOME AXIS #: 7<cr>
LOCATED IN HOME BOUNDS? Y<CR>
WAITING FOR HOME AXIS #007 **PASSED
```

10) At any time issuing the command:

```
>>XREADY AXIS #: 6<cr>
>>XREADY AXIS #: 7<cr>
```

will send the requested gantry axis to the location at which you zeroed it in step 6).

Your robot and gantry system is now ready for operation. Note that the READY and HOME commands are for the Robot arm only, you must use XHOME and XREADY commands for the gantry axes.

Function 04

Description

Access to the RAPL-II variable table is provided through this function level. Data can be removed from, or added to the RAPL-II variable table. Often, it is more efficient to pull data from the table and use it and then return the data.

Common Inputs

AL = 04

ES:[BX] points to a variable table structure, either as a source for data, or as a destination for data extracted from the table. A variable table structure is defined by the following 'C' reference:

```
struct vartable
   {
   char name[8];
   float value;
   int checksum;
   }
```

In no case is the checksum used by the BIOS; this structure is only used to maintain compatibility with RAPL-II.

Subfunction 01

Description

Update the RAPL-II variable table with the new data. If the variable does not exist, it will be created with the new value.

Inputs

AH = 1

Outputs

RAPL-II variable table will be updated.

Subfunction 02

Description

Delete the entry from the table. The input structure doen't need a value, but it must have a proper RAPL-II name to identify the target variable for deletion.

Inputs

AH = 2

Outputs

Selected variable will be deleted.

J-4 USING EXTRA AXES #6, 7 AND 8 TOGETHER (Continued)

The following auxiliary signals are also available on the EXPANSION DC AMPLIFIER connector:

+5VDC @ 800 ma max. . . . . pin #13
+12VDC @ 500 ma max . . . . . pin #14
Signal Ground . . . . . pin #18

Protective Ground(SHIELD) . . . . . pin #8

ARM ON (Amp Enable, LO when arm is on) . . . . . pin #19
BRAKE (24 VDC @ 600ma. HI when arm is on) . . . . pin #20 & 25

The CRS part number for the mating connector shell for the EXPANSION DC AMPLIFIER is R-CPC/RFP039. Inserts for this connector are part number R-CPC/CCP507. Order as many inserts as you need, one for each of the contacts listed above that you require. These parts can be ordered from your CRS Plus distributor.

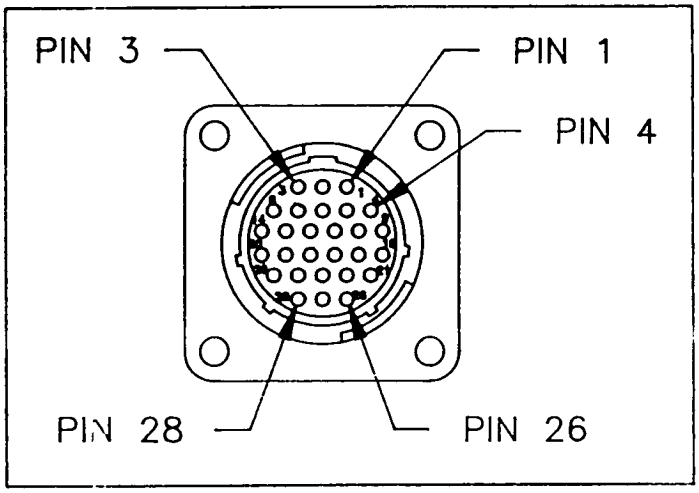


FIGURE J-2 - The extra axis EXPANSION DC AMPLIFIER connector

Subfunction 01

Description

Update the RAPL-II location table with the new data. If the location does not exist, it will be created with the new values.

Inputs

AH = 1

Outputs

RAPL-II location table will be updated.

Subfunction 02

Description

Delete the entry from the table. The input structure doen't need a value, but it must have a proper RAPL-II name to identify the target location for deletion.

Inputs

AH = 2

Outputs

Selected location will be deleted.

Subfunction 03

Description

Return the value from the RAPL-II location table into the local structure. If the variable did not previously exist, then a RAPL-II error will result, and the program will terminate.

Inputs

AH = 3

Outputs

Local structure value will be updated.

Function 06

Description

Reserved for future use.

Inputs

Outputs

J-5 ELECTRICAL CHARACTERISTICS

Internal Amplifier

The sixth DC amplifier has the following output characteristics:

V_{max} +/- 20 VDC

I_{max} +/- 5 Amps Peak Current.

I_c +/- 2 Amps Continues current (using a 2 Amp Thermal circuit breaker)

Optical Encoder Interface

The optical encoder must have three channels (A, B, and Zero) of square wave output with standard TTL level (refer to APPENDIX E of the Robot System Technical manual):

LO < 0.8 Vdc

HI > 2.0 Vdc

Function 09

Description
Route through the standard RAPL-II error recovery sequence. The machine code program will terminate, and the RAPL-II interpreter will be entered after the recovery.

Inputs
AL = 09
CX = RAPL-II error code 0-255

Outputs
None

Function 0A

Description
Analog input/output support.

Common Inputs
AL = 0A

Subfunction 01
Description
Output to analog channels. Analog output channels 1 or 2. Output values are 0-255 (8 bits).

Inputs
AH = 1
CL = output value
CH = channel number

Outputs
None

Subfunction 02
Description
Read analog input channels

Inputs
AH = 2
CH = channel number (9-25)

Outputs
AL = analog input value (0-255)

Function 0D

Description
Process RAPL-II interrupt vectors

Common Inputs
AL = 0D
CL = vector number (0-255)

Subfunction 01
Description
Read the current state of the interrupt value.

Inputs
AH = 1

Outputs
ES:[BX] contains the current interrupt vector value.

Subfunction 02
Description
Enter a new vector value into the vector table.

Inputs
AH = 2
ES:[BX] new vector value.

Outputs
None

Subfunction 03

Description

Perform an axis card communication function. See technical literature for the communication format of the CRS IAXIS axis card. This category of communication functions issues a single command word to the axis card, and a programmable number of data words is returned.

Inputs

AH = 3
CL = Command number (0-255)
CH = Number of data words to return (extra status word is appended, and not included in this count)
ES:[BX] - destination of return data

Outputs

ES:[BX] will point to a valid data block. If an error occurs, a general RAPL-II error recovery will bw initiated.

Subfunction 04

Description

Transmit a word string to the axis card

Inputs

AH = 4
ES:[BX] points to an input word string
CX = number of words to transmit

Outputs

None

J-1 INTRODUCTION

The A100/200 robot controller is designed to control up to 8 servo axes. The controller electronics supports these axes by expansion "slots" on the mother board. The robot arm uses 5 of these axes. If a CRS servo or magnetic gripper is used by the system, the eighth slot is used by the gripper controller card. Any other slots can be used to control servo axes to operate external servo devices.

Examples of external servo axes are:

- A robot track upon which the arm is moved,
- A gantry structure which carries the arm,
- A servo rotary table to position parts or fixtures,
- A rotary welding positioner, or
- A conveyor monitoring system (an encoder used to monitor conveyor speed and/or position.

The most common forms of external axes are the track and gantry. For this reason RAPL-II also supports these two formats in specialized software. Other external axes can be controlled and coordinated with general external axis commands.

This APPENDIX documents the set up of three types of extra axis set-up. The first is the robot track. The robot controller includes an extra amplifier which can be used to power a servo motor of similar size and performance to the ones used in the robot arm (ME-2110). A track powered by such a motor can drive the standard robot arm at reasonable speed, acceleration rate, and accuracy. The amp is built in but special cables are required to take the encoder and motor power signals to the external motor.

The second is a gantry system also based on the M2110 motors. In this case, an external amplifier must be included in the system.

Thirdly there is the general case where a third party amplifier/motor/encoder is being used. The final section of this APPENDIX describes the interface requirements for this type of installation.

J-2 USE OF EXTRA AXIS #6 WITH A TRACK (Continued)

Software Set up:

When powering up a controller that has just had a 6th axis card added, or whenever a TEACH START is done on a controller having more than 5 axes installed, the controller will echo this message before the sign-on message:

EXTRA CARD DETECTED, ENTER PROPER NUMBER OF AXES:_

The user must enter the correct number of axes installed. In this case, enter 6<cr>. The sign on message will then appear on the screen.

To configure the system, the following steps must be taken. For complete detail about these commands, refer to APPENDIX A of the Technical Manual ("@"-commands) and the RAPL-II programming manual (other commands):

1) Issue the PASSWORD. Refer to the letter in the front of your manual package for this number.

2) Issue the command to set the number of pulses per encoder revolution for the new servo axis. Without this, RAPL-II will not know how many pulses to issue per motion command. To do this, use the @XPULSES command:

   >>@XPULSES AXIS #: 6, ENCODER PULSES/REV = 01000/ppppp<cr>

   Where ppppp is the number of pulses per one revolution of the encoder.

3) Enter the conversion factor from encoder turns to command units. Command units are units which will be used in commands such as JOINT. Once set up, commanding "JOINT 6,12" will cause the track to move 12 inches.

   >>@XRATIO AXIS #: 6, MOTOR/JOINT RATIO = 001.0000/rrr.rrrr<cr>

   Where rrr.rrrr is the conversion factor from MOTOR TURNS to UNITS OF MEASUREMENTS. For example, if a track has a gear ratio that will cause a motion of 1" per 2.5 turns of the motor, then rrr.rrrr is equal to 2.5

4) Enter the joint limits. These are the positive and negative limits of the joint travel in commanded units.

   >>@XLIMITS AXIS #: 6, POSITIVE JOINT LIMIT: +LIM, NEGATIVE JOINT LIMIT: -LIM<CR>

   Where +LIM is the positive travel limit (from the location were the track will be zeroed) in command units. And where -LIM is the negative travel limit (from the location were the track will be zeroed) in command units.

J-2 USE OF EXTRA AXIS #6 WITH A TRACK (Continued)

5) Move the track to the location where all locations will be measured from. This is the "zero" for the track.

6) Enter the following command:

```
>>XZERO AXIS #: 6<cr>
```

7) Now move the track to the location from which it must home. This could be a limit switch at one end of travel, or a known "docking" location from which the track will start each time the system is started. At this location (the homing location) issue the following:

```
>>XCAL AXIS #: 6<cr>
LOCATED IN HOME BOUNDS? Y<CR>
WAITING FOR HOME AXIS #006 **PASSED
```

8) To use a limit switch for a "hard" homing location, use the following code to move the track to the switch:

```
PROGRAM MOVE_LIM:                  Call by GOSUB MOVE_LIM 6,SW6.
1000 ENABLE SLEW
1100 SPEED DEADSLOW                Try setting DEADSLOW to 10%.
1200 JOINT %0,-1000                Use a distance beyond the track limit ie. 1000 inches!
1300 WAIT %1                       SW6 is the input number used for the limit switch.
1400 HALT
1500 DISABLE SLEW
1600 XHOME %0                      For set-up, STOP before this line.
1700 RETURN
$
```

If this is used for a 7th or 8th axis, the GOSUB call will be:

```
GOSUB MOVE_LIM 7,SW7, or GOSUB MOVE_LIM 8,SW8
```

9) At this time you may apply homing marks to indicate the range of travel within which the track must be before initializing the Homing procedure. After the Track has stopped with the message above, it is at one zero cross or index mark on the encoder. The encoder must be within one turn of this location before homing. Make a pointer on the moving part of the track to indicate track position. Where it points to now, make a mark on the stationary track component. Issue the command:

```
>>MOTOR 6,-nnn<cr>
```

J-3 USING EXTRA AXES #6 AND 7 AS A GANTRY

Hardware Considerations:

If axis 6 and 7 are used for a gantry using MCS-M2110's or similar motors, the CRS Plus 3-Channel DC Amplifier (DCA3) can be used (output specifications are found in section J-5). To use the extra axes, an axis card either P-Type or PID-Type (A150 or A250) must be installed in slots numbered 6 and 7.

The following parts are required for inter-connecting the GANTRY/CONTROLLER/ AMPLIFIER system.

Quantity	Description	Part Number
1	DCA3 - A100/200 Series DC Amplifier	SEC-13-033
2	M2110 - Motor/Encoder set.	SEC-09-029
2	M2110CXX - Motor Cables (XX is 10, 25, or 40 Foot)	SCC-09-024/026/044
1	RARM/CXX - Robot Arm Cable (XX is 20, 30, or 40 foot)	SCC-14-520/521/522

Software Installation:

When powering up a controller that has just had extra axis cards added, or whenever a TEACH START is done on a controller having more than 5 axes installed, the controller will echo this message before the sign-on message:

EXTRA CARD DETECTED, ENTER PROPER NUMBER OF AXES:_

The user must enter the correct number of axes installed. In this case, enter 7<cr>. The sign on message will then appear on the screen.

The installation of the Gantry software is the same as for the Track as described in section J-2 with one exception: the @GANTRY command is used in place of the @TRACK command. All set-up commands must be used for both axes 6 and 7. For complete detail about these commands, refer to APPENDIX A of the Technical Manual ("@"-commands) and the RAPL-II programming manual (other commands). The sequence used for each axis is:

1) Issue the PASSWORD.

2) Issue the command to set the number of pulses per encoder revolution for each new servo axis.

```
>>@XPULSES  AXIS #: 6, ENCODER PULSES/REV = 01000/ppppp<cr>
>>@XPULSES  AXIS #: 7, ENCODER PULSES/REV = 01000/ppppp<cr>
```

3) Enter the conversion factor from encoder turns to command units.

```
>>@XRATIO AXIS #: 6,  MOTOR/JOINT RATIO = 001.0000/rrr.rrrr<cr>
>>@XRATIO AXIS #: 7,  MOTOR/JOINT RATIO = 001.0000/rrr.rrrr<cr>
```

J-3 USING EXTRA AXES #6 AND 7 AS A GANTRY (Continued)

Gantry Calibration and Homing:

Use the following procedure is used to calibrate the gantry:

1) Turn on the MAIN AC power

2) Turn on the ARM power

3) Enter the PASSWORD

4) Put the system in a MANUAL mode and use the switches labelled AXIS 6 and 7 to move the gantry. Be careful at this time not to collide with the ends of travel as the limits are not checked.

5) Move each axis of the gantry to the locations where all gantry positions will be measured from. This is the "zero" for the gantry.

6) Enter the following commands:

```
>>XZERO AXIS #: 6<cr>
>>XZERO AXIS #: 7<cr>
```

7) Now move each axis of the gantry to the location from which it must home. This could be a limit switch at one end of travel, or a known "docking" location from which the gantry axes will start each time the system is started. For use with a limit switch refer to Section J-2. At this location (the homing location) issue the following:

```
>>XCAL AXIS #: 6<cr>
LOCATED IN HOME BOUNDS? Y<CR>
WAITING FOR HOME AXIS #006 **PASSED
>>XCAL AXIS #: 7<cr>
LOCATED IN HOME BOUNDS? Y<CR>
WAITING FOR HOME AXIS #007 **PASSED
```

8) At this time you may apply homing marks to indicate the range of travel within which the two gantry tracks must be before starting the Homing procedure. After the Track has stopped with the message above, it is at one zero cross or index mark on the encoder. The encoder must be within one turn of this location before homing. Make a pointer on the moving part of each track to indicate its position. Where it points to now, make a mark on the stationary track component. Issue the commands:

```
>>MOTOR 6,-nnn<cr>
>>MOTOR 7,-nnn<cr>
```

J-4 USING EXTRA AXES #6, 7 AND 8 TOGETHER

Hardware Considerations:

1) Axis 6,7 and 8 using DCA3 Expansion Amplifier:

When using all three expansion axes 6, 7 and 8 in a user application, three axis cards (P-Type or PID-Type depending on whether the system is a A150 or A250) must be installed in slot numbers 6, 7 and 8. The MOTOR 8/SERVO GRIPPER switch in the back of the controller must be set to MOTOR 8.

If MCS-M2110 or similar motors are to be used, the 3-channel DC Amplifier package can be used. If so, the following parts are required for interconnecting the CONTROLLER-AMPLIFIER-ROBOT system:

Quantity	Description	Part Number
1	DCA3 - A100/200 Series DC Amplifier	SEC-13-033
3	M2110 - Motor/Encoder set.	SEC-09-029
3	M2110CXX - Motor Cables (XX is 10, 25, or 40 Foot)	SCC-09-024/026/044

2) General Data for Any Other Amplifier Installation

If larger size motor/encoder sets than the M2110 are required (see Section J-5 for specifications), then an external DC or AC SERVO AMPLIFIER can be used. The following input/output signals are available on the EXPANSION DC AMPLIFIER connector (see Figure J-2):

OUTPUT:

Vcom 6 is a +/- 10 VDC @ 20 ma. . . . . pin #4
Vcom 7 is a +/- 10 VDC @ 20 ma. . . . . pin #12
Vcom 8 is a +/- 10 VDC @ 20 ma. . . . . pin #21

INPUT:

ENCODER channel 6A . . . . . pin #1
ENCODER channel 6B . . . . . pin #2
ENCODER channel 6Z . . . . . pin #3

ENCODER channel 7A . . . . . pin #9
ENCODER channel 7B . . . . . pin #10
ENCODER channel 7Z . . . . . pin #11

ENCODER channel 8A . . . . . pin #22
ENCODER channel 8B . . . . . pin #23
ENCODER channel 8Z . . . . . pin #24

J-4 USING EXTRA AXES #6, 7 AND 8 TOGETHER (Continued)

Software Installation:

When powering up a controller that has just had extra axis cards added, or whenever a TEACH START is done on a controller having more than 5 axes installed, the controller will echo this message before the sign-on message:

EXTRA CARD DETECTED, ENTER PROPER NUMBER OF AXES:_

The user must enter the correct number of axes installed. In this case, enter 8<cr>. The sign on message will then appear on the screen.

The installation of the extra axis software is the same as for the Gantry software as described in section J-2 with two exceptions: do not issue the @TRACK or @GANTRY commands and expand all set-up commands to 8 axes from 6. For complete detail about these commands, refer to APPENDIX A of the Technical Manual ("@"-commands) and the RAPL-II programming manual (other commands).

The following commands must be used for each of axes 6, 7, and 8:

@XPULSES
@XRATIO
@XLIMITS
@XMAXVEL
XCAL

XHOME - before each use of the external axis.