Techno Inc.

Wiring, Setup, & ASCII Programming For C-Series Controllers

(IT116G, C10, C116 & C142)

Techno, Inc.
2101 Jericho Turnpike, Box 5416
New Hyde Park, NY 11042-5416
Phone: (516) 328-3970
Fax: (516) 358-2576
www.techno-isel.com
support@techno-isel.com

HTM00430301

Techno
Wiring, Set-up, And Programming
For C-Series Controllers

Statement of Limitation of Warranties and Liability

Table of Contents

LIST OF FIGURES

1

SETTING UP C-SERIES CONTROLLERS

How To Use This Manual

Congratulations on your purchase of a Techno C-Series Controller. The unit comes fully assembled and consists of the following:

Integrated driver and Interface card (IT116 G only)
2 or 3 stepper drivers (2A for C10; 3.6 A for C116; 6A for C142)
1 Techno Interface Card (4.0 I/O for C10 and C116; 5.0 I/O for C142)
1 Logic Power Supply
1 Enclosure (C10, C116, C142)

The Isel C-Series Controller is a microprocessor-based stepper motor controller. The controllers can control 3 axes (x, y, and z) with linear interpolation available on any 2 axes (with the C10, C116, or C142) or all 3 axes (only with the C142). The IT116G is a single-axis controller. Motion commands need only specify displacements and speeds. The software on the Interface card calculates and performs all acceleration and deceleration ramps from the incoming command data.

The controller can be used with various mechanical systems (for example an x-y table), and a personal computer (PC) to form automated manufacturing systems.

Commands and programs can be loaded directly from a dumb terminal or through a software development system which runs on IBM-PC compatibles. Communications are through a serial bus. Programs loaded into the controller can be executed directly or stored. Once a program is stored in the controller, the PC is only necessary for modifying the program and can be disconnected if desired.

This manual gives the basic information you need to use your Isel C-Series Controller.

- All users should read the first section of the manual. This section gives instructions for setting up the controller, including important guidelines on wiring and the use of motor cables, and information on connecting input/output (I/O) devices. This section also tells you how to run a Self-Test Program which you should run immediately after you set up the equipment in order to test the controller and the mechanical system. This is followed by information on downloading programs and communicating with the controller.
- The second section of the manual gives a series of example programs, with explanations, which you may find useful to quickly further your understanding of the controller's functions. You may also find these programs useful as they are for your purposes.
- The third section explains the commands used in programming the controller's Interface Card . Information is given for commands in both immediate and program modes.
- The fourth section is a troubleshooting guide. It indicates some problems you might encounter when first using your controller, along with some steps to isolate, and possibly remedy, the difficulties.
- The fifth section gives reference material for the controller. It gives the error messages and their code numbers, specifications for the Interface Card, listings of important programs, and information on hexadecimal numbers.

Technical Support

Please contact your local dealer or Techno for technical assistance.

- ***By Phone:*** Call us at ***(516) 328-3970***, and specify that you need technical support. It is helpful if you have all the necessary necessary background information ready before calling.

- ***By Fax:*** Send your questions and detailed background information to us by fax at ***(516) 358-2576***. Please remember to include both your phone number, fax number and e-mail address on the transmittal.

Conventions Used In This Manual

The following conventions are used in this manual.

Italic type is used to highlight terms that are defined or referred to in the manual.

Bold italic type emphasizes important information within the text.

Text that appears on your computer screen or that needs to be input by you is written in this print.

☞ ***This symbol appears to the left of a note containing important information related to the text.***

✋ ***THIS SYMBOL APPEARS TO THE LEFT OF A WARNING REGARDING THE POSSIBILITY OF EQUIPMENT DAMAGE IF CERTAIN GUIDELINES ARE NOT OBSERVED.***

Equipment

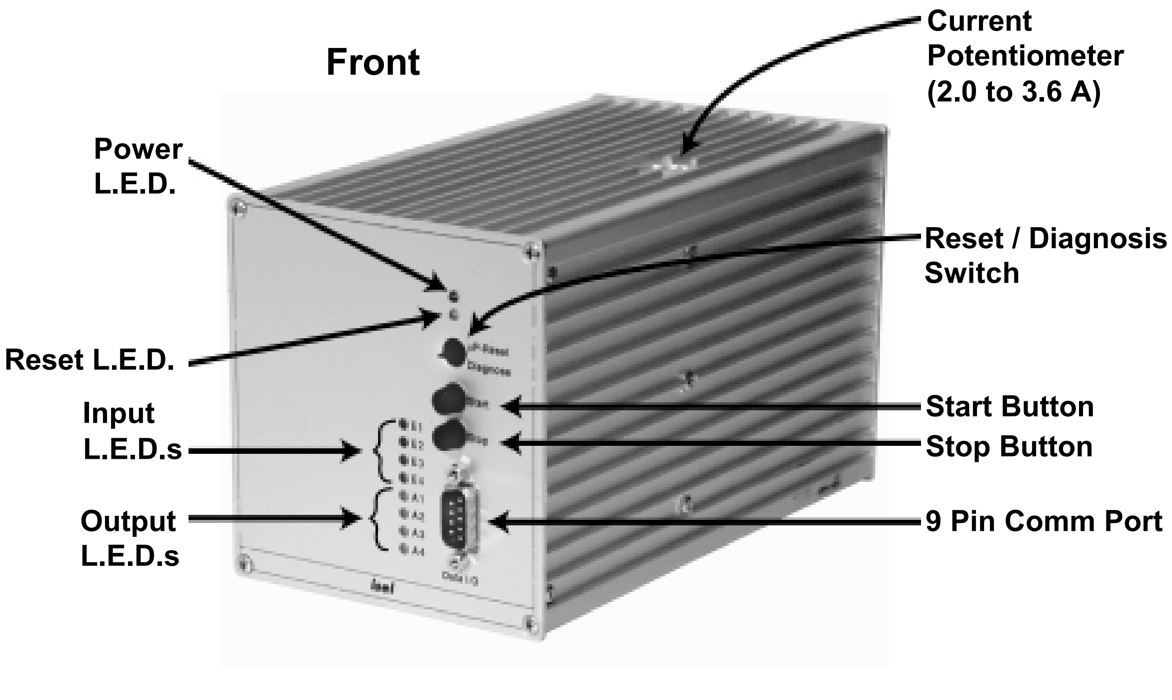
Examine the contents of your box to make sure you have all the equipment needed to set up your Isel C-Series Controller. This should include:

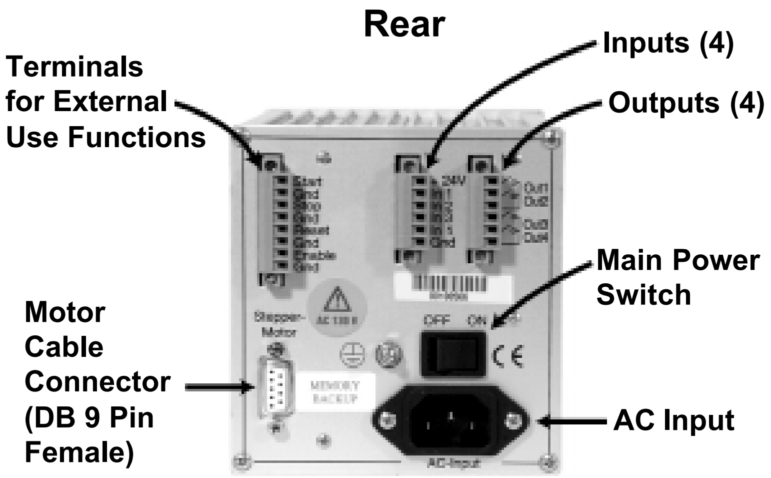
IT116G, C10, C116, or C142 enclosure (pre-assembled as described on page 1)
9 to 9 pin RS232 serial cable
Power cord
1, 2 or 3 motor cables (depending on controller model)
1 Techno Linear Motion Software & Manuals CD-ROM

If you are missing any of the above equipment, contact Techno **immediately.**

Controller Box Identification Callouts

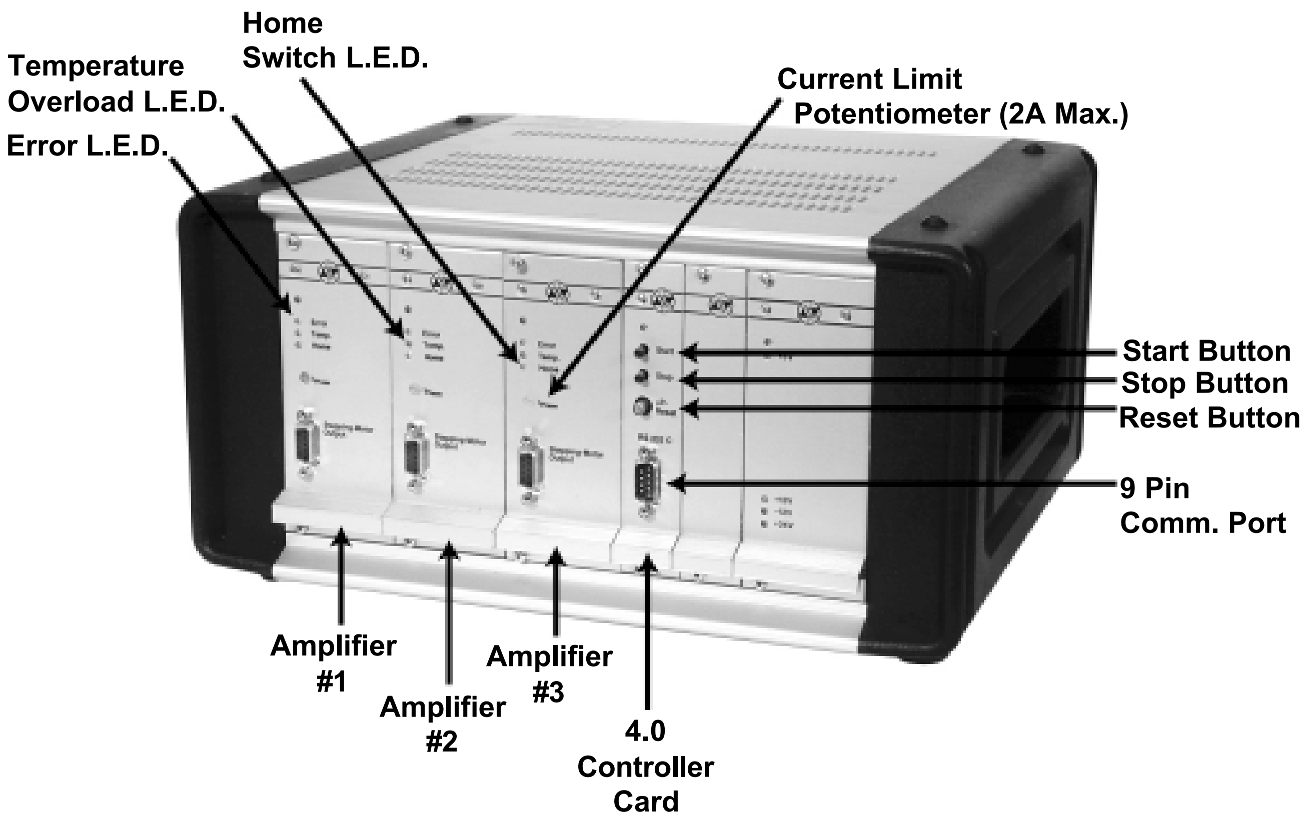
IT116G Controller



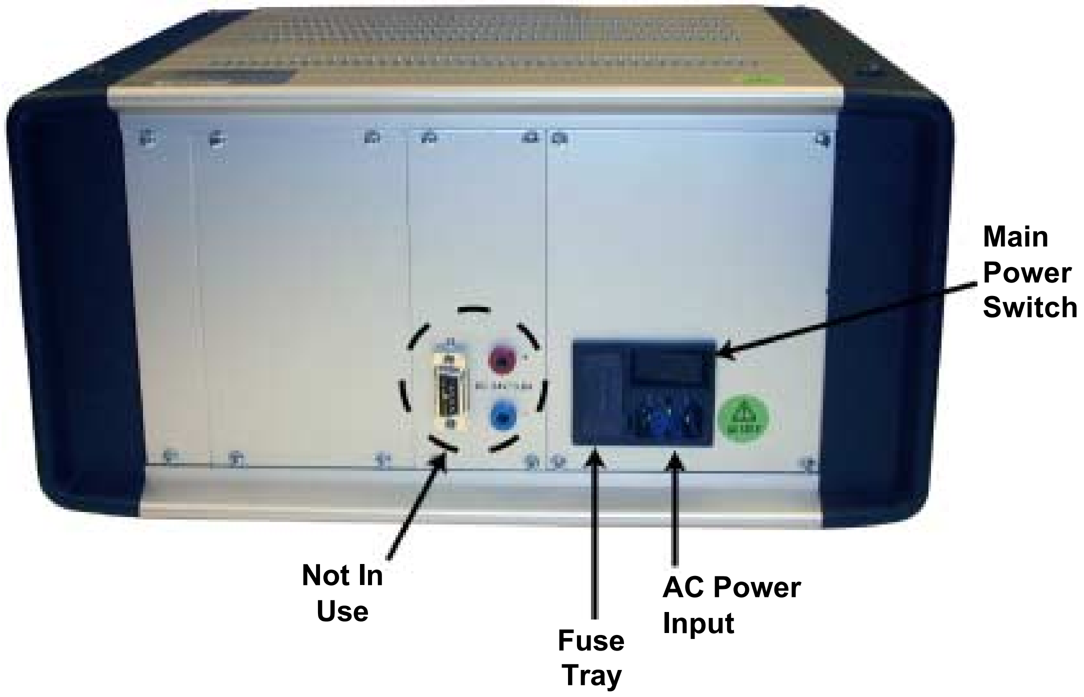

Rear
Inputs (4)
Terminals for External Use Functions
Outputs (4)
Main Power Switch
Motor Cable Connector (DB 9 Pin Female)
AC Input

C10 Controller

Front

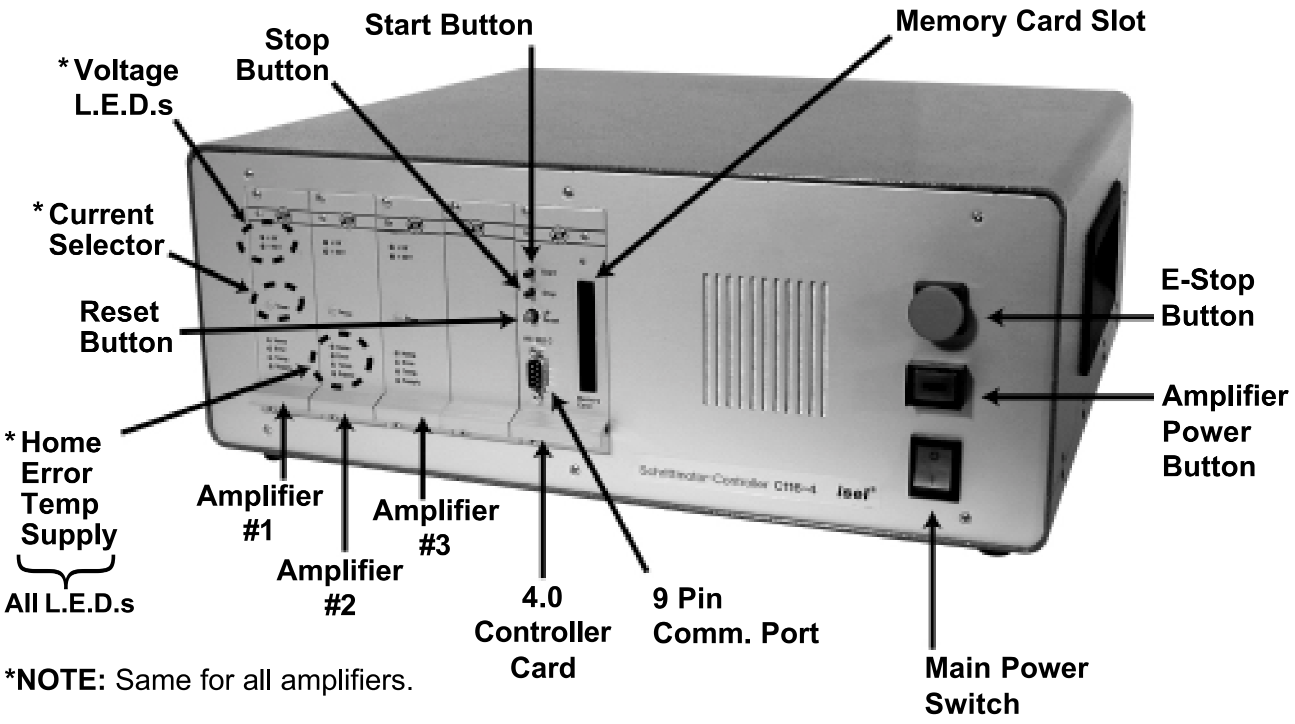


Rear



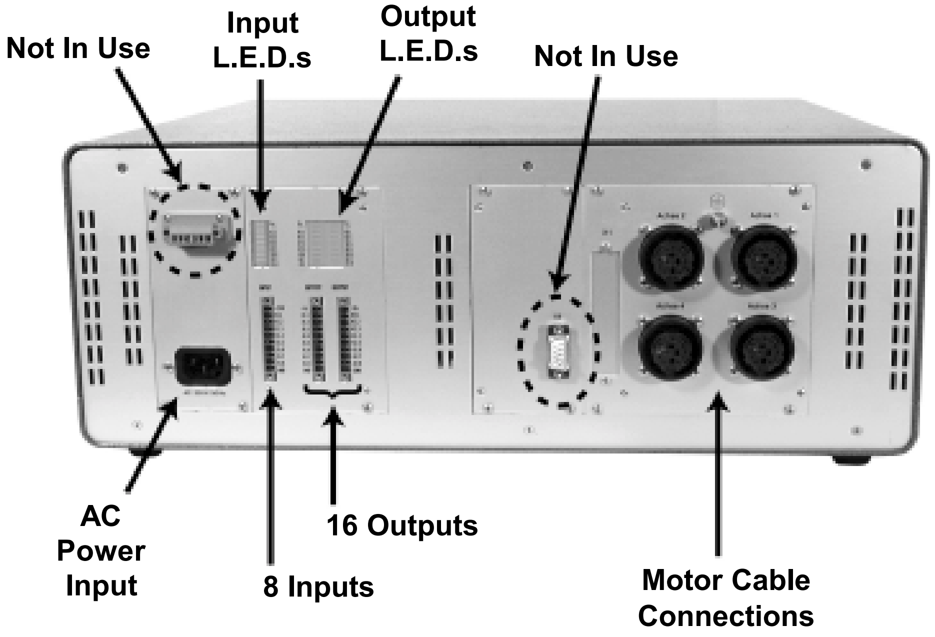
C116 Controller

Front



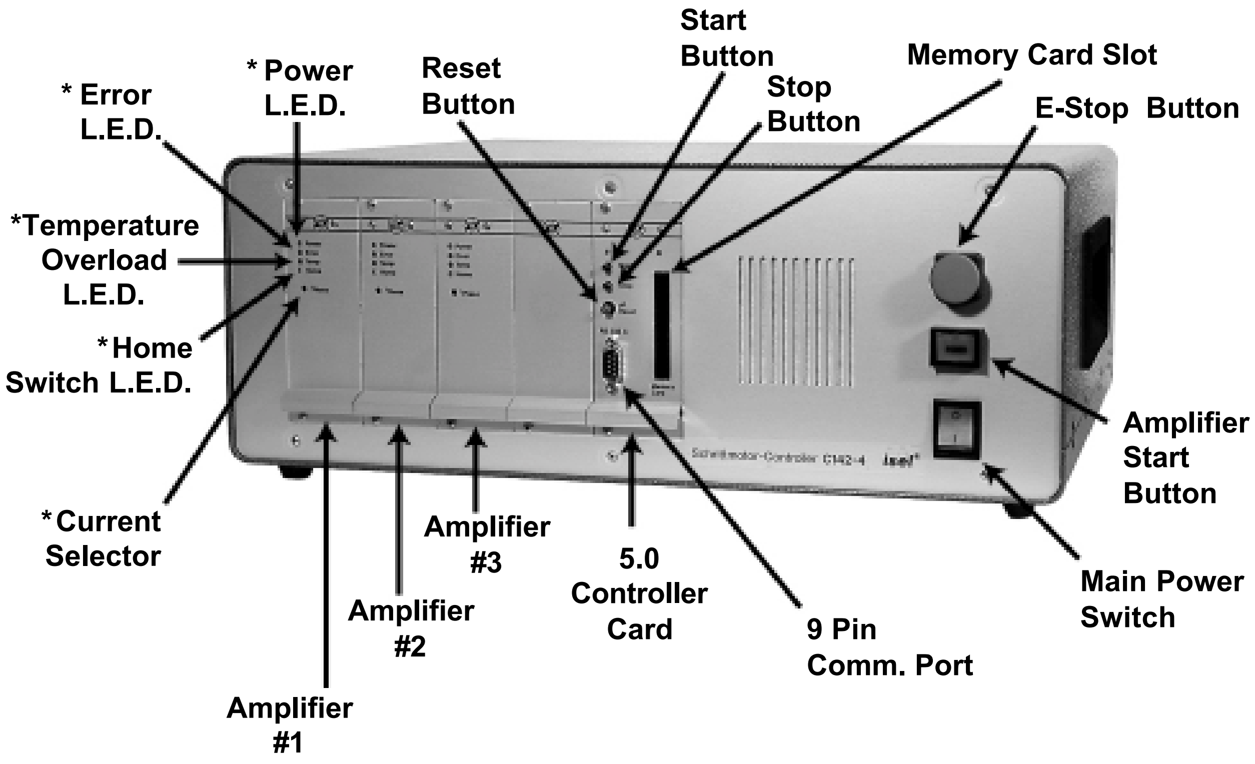
*NOTE: Same for all amplifiers.

Rear



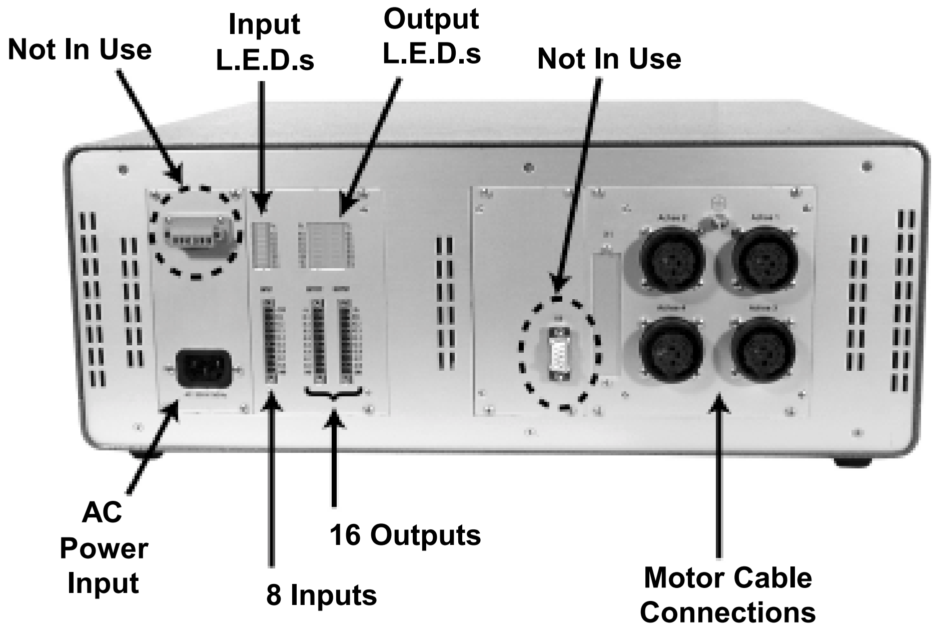
C142 Controller

Front



***NOTE:** Same for all amplifiers.

Rear



Description of Controller Callouts

Main Power Switch – Turns AC power on or off to the unit.

Amplifier Power Button – Allows amplifiers to be energized.

Start Button – Momentary button to start program residing in controller's memory.

Stop Button – Momentary button to pause program residing in controller's memory. It stops the program, even in the middle of a move and waits for instructions in the form of reset button or start button.

E-Stop Button – Shuts down power to the entire controller including logic signals.

Reset Button – Ends execution of program. If start button is pressed, program starts at beginning. Does not wipe memory clean, if Battery Back Memory/Memory Card used.

Reset/Diagnosis Switch – Only on the IT116G. Ends execution of program. If start button is pressed, program starts at beginning. Does not wipe memory clean. Diagnosis switch sends error codes to comm. port.

Amplifier Card – Electronics card that provides power to a single step motor.

Memory Card Slot – Accepts Isel 32K Memory Card slot upon which programs can be stored.

Current Potentiometer – Selects the amount of current (A) supplied to the step motor. Max. current is 3.6A/phase on IT116G; 2A/phase for C10; 3.6A/phase for C116; 6A for C142.

Power LED – Indicates power is supplied to unit or amplifier.

Reset LED – Indicates reset button is pressed. No holding torque to motors.

Input LED – Indicates input "x" has been activated.

Output LED – Indicates output "x" has been activated.

Error LED – Indicates error has occurred with the particular axis. See error codes on page 90.

Temperature LED – Indicates overload of amplifier. That particular axis has no power to it.

Home LED – Indicates limit switch is engaged or circuit is open. Unit will not run until this condition has been cleared.

AC Power Input – Power cord is plugged in here. 120VAC or 230VAC depending on model.

9 pin comm. Port – Null modem communications port. Must use Null modem cable or you may potentially damage computer's serial port.

Inputs – Digital inputs which can be programmed through the comm. port (See page 15 for wiring information).

Outputs – Digital Outputs which can be programmed through the comm. port (See page 14 for wiring information).

Terminals for External use Functions – Wiring terminal for external use of functions such as Start, Stop and Reset.

Motor Cable Connections – Motors cables are attached here and other end goes to motor housing. DB9 connectors are used for IT116G and C10. Amphenol connectors are used for C116 and C142.

Controller Card 4.0 - Main logic card for the C10, C116 controller.

Controller Card 5.0 - Main logic card for the C142 controller.

Fuse Tray – Holds fuse for the C10 controller.

Factory Jumper and/or DIP Switch Settings

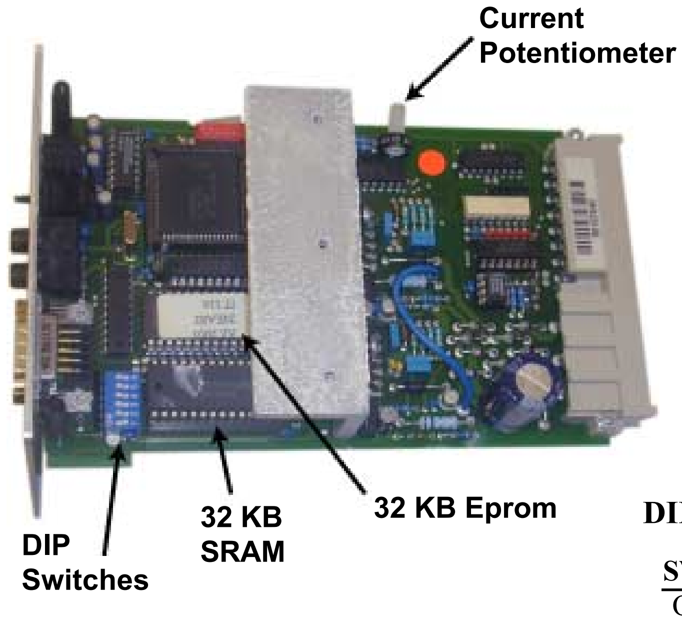
These factory settings are appropriate for the purposes of most users. If they do not serve your purposes, you should adjust the settings on the card as needed. The following diagram of the Interface Card is provided to assist you.

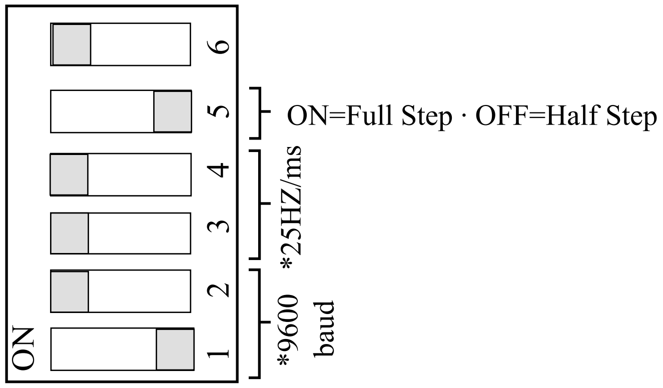
☞ ***If you are using your controller with a Techno slide or table you will not need to adjust any settings. Skip this section and proceed to the section on Attaching Motor Cables.***

ONLY THE MOST EXPERIENCED USERS SHOULD MAKE ADJUSTMENTS TO THE INTERFACE CARD. IF YOU ARE UNSURE ABOUT MAKING SUCH ADJUSTMENTS, CALL TECHNO. IMPROPER JUMPER SETTINGS WILL CAUSE THE CONTROLLER TO FUNCTION IMPROPERLY.

IT116G Controller Card DIP Switch Settings

This section gives the jumper settings on the IT116G Controller Card (not applicable to Interface Card 4.0 and Interface Card 5.0).





*default setting

DIP SWITCH SETTINGS

SW1	SW2	Baud Rate
ON	OFF	4800
OFF	ON	9600*

SW3	SW4	Accel .Rate
ON	ON	25HZ/ms*
OFF	ON	50HZ/ms
ON	OFF	75HZ/ms
OFF	OFF	100HZ/ms

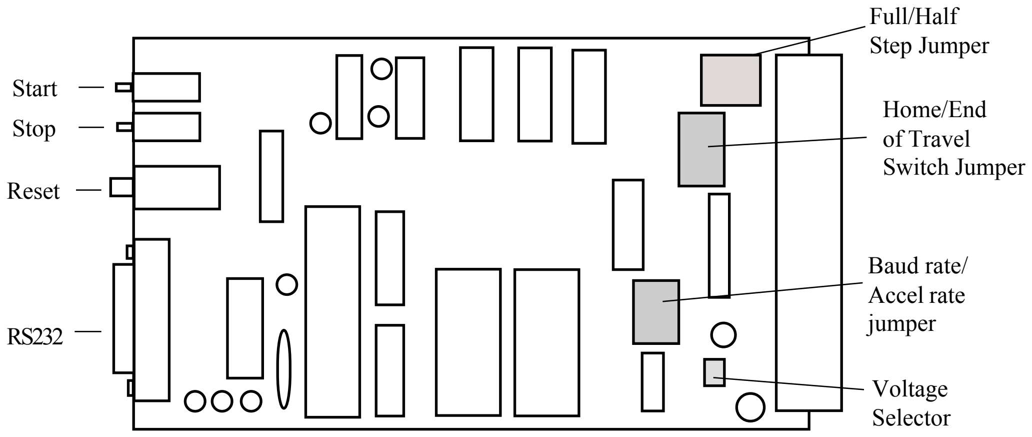
SW5	-	Step Size
ON	-	Full Step
OFF	-	Half Step*

*default setting

C Series 4.0- 5.0 Interface Card Jumper Settings

This section gives the jumper settings on Interface Card 4.0 and Interface Card 5.0 (not applicable to IT116G). There are four important jumper blocks on the card which are set as follows:

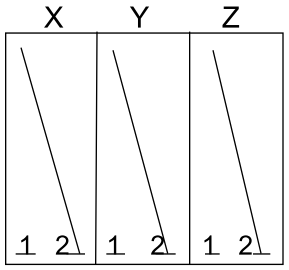
Jumper Block	Factory Setting
Full /Half Step	Half step for all axes
Home/End of Travel Switch	Home Switches Activated
Voltage Selector	+5V
Baud Rate/Acceleration rate	9600 Baud/75 Hz/ms

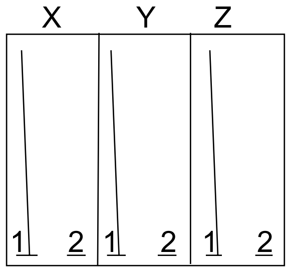


Start
Stop
Reset
RS232
Full/Half Step Jumper
Home/End of Travel Switch Jumper
Baud rate/ Accel rate jumper
Voltage Selector

Full Step/Half Step Jumper

This jumper determines whether typical motion will be in full steps or half steps. The jumper consists of three bars, one for each axis. If a bar is attached to the side labelled "1", then that axis is in full step motion. If the bar is attached to the side labelled "2", then that axis is in half step motion. See diagrams below:



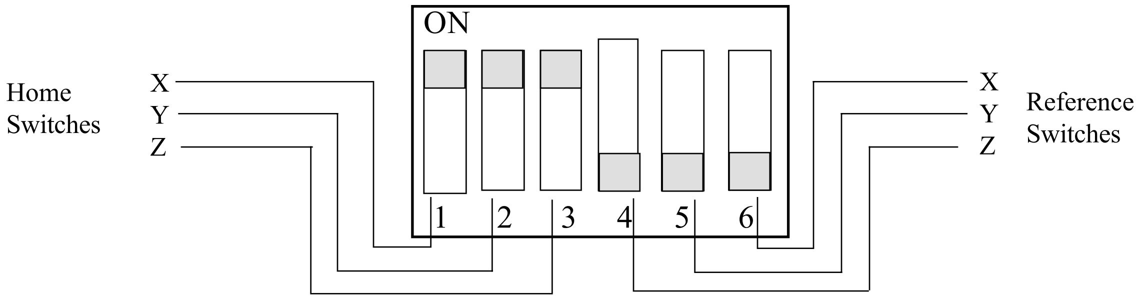


All Axes in Half Step Mode (Default) **All Axes in Full Step Mode**

We do not recommend full step motion, since this generally produces excessive vibration.

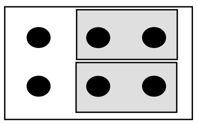
Home/End Of Travel Switch Jumper

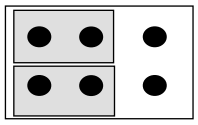
This jumper activates the home (limit) switches and the reference switches. Switches 1, 2, and 3, are the home switches for the X, Y, and Z axes respectively. Switches 6, 5, and 4 are the reference switches for the X, Y, and Z axes respectively. The default setting (illustrated below) activates all three home switches while making all reference switches inactive (because Techno tables do not normally use the reference switches). **Note that the side labelled ON is the inactive side.**



Voltage Selector

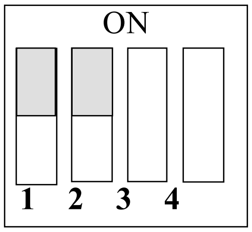
This jumper sets the voltage for drivers. The default setting is +5V.


+5V (Default)

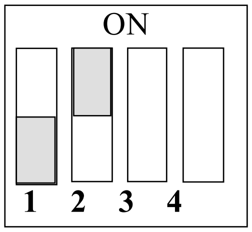

+12V

Baud Rate/Acceleration Rate Jumper

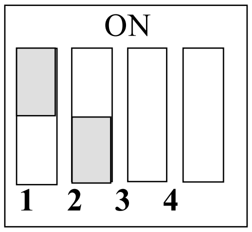
The same jumper is used to adjust the baud rate and the acceleration rate. The positions of switches 1 and 2 set the baud rate for communication. The default is 9600 Baud. This and the other possible settings are illustrated below:



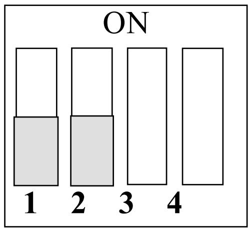
19200 baud



9600 baud (default)

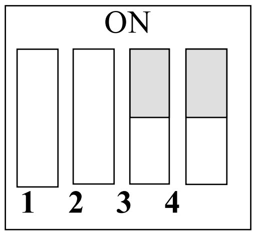


4800 baud

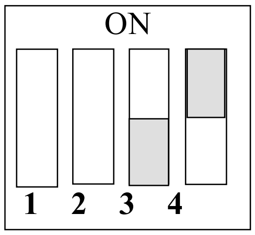


2400 baud

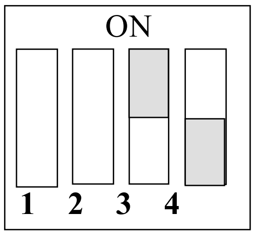
The positions of switches 3 and 4 set the acceleration rate for all motors. The default is 75Hzms. This, along with the other possible settings, is illustrated below:



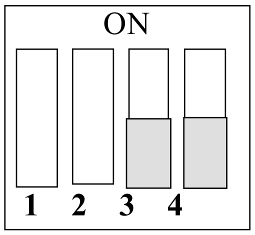
25HZ/ms



50HZ/ms



75HZ/ms



100HZ/ms

Once all jumpers are set, you can proceed with the setup of the controller

Use the diagrams on the previous pages, showing the rear of the controller, as a guide for attaching motor cables and hooking up I/O devices as described in the following pages.

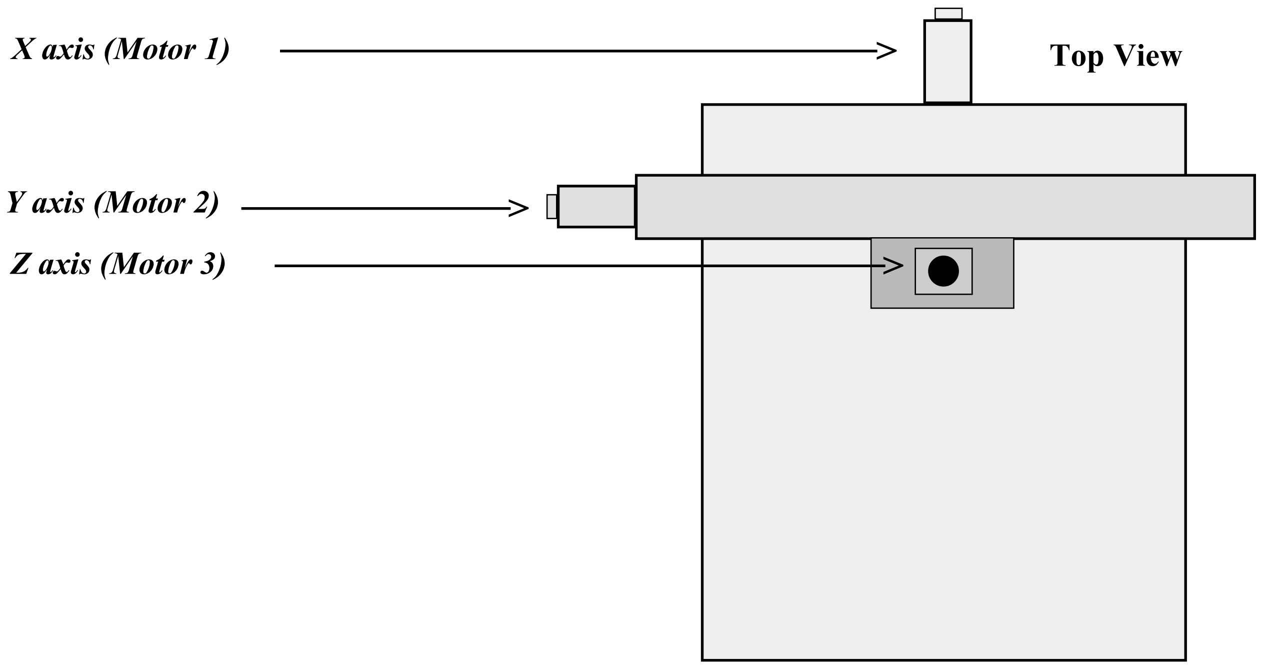
Attaching Motor Cables

1. Place the controller in close proximity to the machine.

MAKE SURE THAT THE CONTROLLER IS SWITCHED OFF WHEN CONNECTING AND DISCONNECTING CABLES. REMOVING A CABLE WITH THE POWER ON TO THE CONTROLLER CAN DAMAGE THE ELECTRONICS.

2. Connect each motor cable by attaching the male end to the appropriate output axis connector on the controller's rear panel (see previous pages), and the female end to the corresponding axis/motor (see diagram below). The cables should be hooked up as follows:

Connect Cable from:	Output X or 1-Axis	to	X-axis (Motor 1) of table
	Output Y or 2-Axis	to	Y-axis (Motor 2) of table
	Output Z or 3-Axis	to	Z-axis (Motor 3) of table



The motor cables are arranged in this way only if you are using the controller with a Techno Slide Table. If you are not using the controller with a Techno table, then you may designate each motor as you wish, or according to the configuration of the mechanics.

When connecting the DB9 cables, be sure to plug the end of the cable into the socket **fully**, and then **tighten** the thumb screws on either side or else the unit may short out causing damage to the electronics as well as the motors.

Connecting the Serial Cable

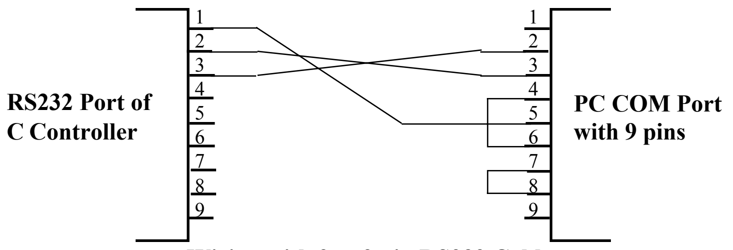
Attach the appropriate end of the RS232 Serial cable to the RS232 Interface on the front of the controller (see pictures on pages 4 thru 4.3). Attach the other end of the Serial Cable to your PC at COM Port 1 or 2.

- ***If your PC has a 25 pin COM Port, you will need a 9 to 25 pin RS232 converter .***

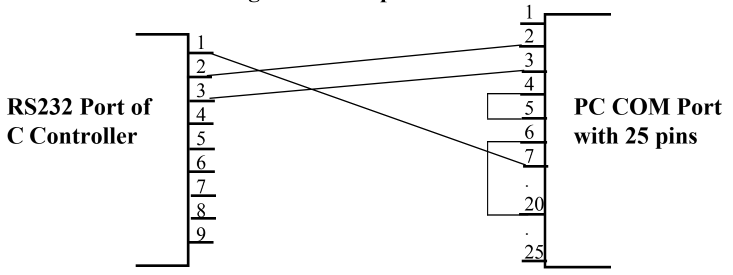
- ***If your PC has a 9 pin COM Port, use the 9 to 9 pin cable enclosed.***

THE SERIAL CABLE HAS ONE END SPECIALLY MARKED FOR THE PC (PC/AT). THE OTHER END IS FOR THE CONTROLLER. DO NOT REVERSE THE CABLE OR DAMAGE MAY RESULT TO THE CONTROLLER AND/OR THE COMPUTER.

The following diagrams illustrate the correct wiring setup between the controller and the PC **as provided by the enclosed** (9 to 9 pin only) **specially wired RS232 serial cables:**



Wiring with 9 to 9 pin RS232 Cable



Wiring with 9 to 25 pin RS232 Cable

IT IS ESSENTIAL THAT THE ENCLOSED RS232 SERIAL CABLE IS USED TO PROVIDED THE CORRECT WIRING BETWEEN THE CONTROLLER AND THE PC. UNDER NO CIRCUMSTANCES SHOULD STANDARD RS232 CABLES BE USED FOR COMMUNICATION BETWEEN THE COMPUTER AND THE CONTROLLER. THIS WILL CAUSE DAMAGE TO EITHER THE COMPUTER, THE CONTROLLER, OR BOTH.

The serial controller makes use of standard circuits and should operate with most PC's.

The serial controller makes use of standard circuits and should operate with most PC's.

The computer should be configured as a DCE device.

The communication parameters require the following settings:	9600 Baud
	8 Bits
	1 Stop bit
	No Parity

Guidelines For Motor Cables And Wiring

WHEN SETTING UP YOUR CONTROLLER, PAY PARTICULAR ATTENTION TO THE FOLLOWING GUIDELINES. FAILURE TO ADHERE TO THESE MEASURES CAN RESULT IN DAMAGE TO YOUR CONTROLLER.

Motor Cables

We recommend 10 foot, 20 AWG motor cables.

- The **absolute maximum length** for a motor cable is 25 feet with 16AWG only.
- **Do not join two cables with a connector to form a longer cable.** The connector forms a bad contact and will cause damage and/or erratic performance during operation of your machine and your controller.
- **DO NOT USE STANDARD STORE-BOUGHT RS232 CABLE FOR MOTORS.** This cable is usually only 24 AWG and will heat up during use causing the motors to malfunction.
- **Do not tie wrap power cord from spindle motor with any motor cable.** The motor cable will pick up spikes from the AC power line and this will produce electrical noise in the controller, causing the machine and the controller to perform erratically.

Wiring and Grounding

- **Do not use the same outlet for the spindle motor and the controller.**
- Never run AC lines strapped to signal lines or motor cables.
- **If you use shielded wire,** make sure the shield is connected only at the controller end.
- **When hooking up input and output devices**, make sure you have one central ground point for all the I/O connections.

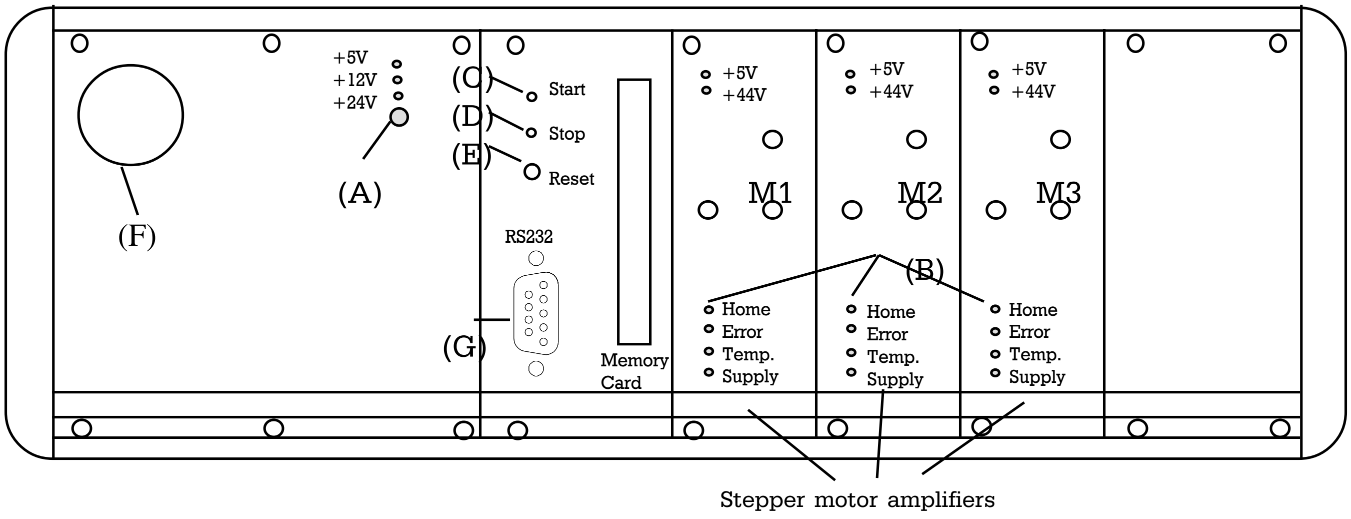
Placement Of The Amplifiers

Note that Techno ships the controllers with the amplifiers already installed. This section is provided as a reference only. No installation of the amplifiers is required.

The C-Series Controllers have slots for up to 3 stepper motor amplifiers.

- The C108 and C116 Controller use 4.3A amplifiers
- The C142 Controller uses 6A amplifiers

Each stepper motor amplifier should be placed in the slot designated by the required motor as shown in the following diagram. If you are using the machine for axes X, Y, and Z, then use motor slots M1, M2, and M3 respectively.



To remove an amplifier:

BE SURE TO TURN OFF POWER TO THE CONTROLLER BEFORE REMOVING ANY AMPLIFIERS. FAILURE TO DO SO WILL RESULT IN DAMAGE TO THE AMPLIFIERS AND TO THE CONTROLLER.

1) Remove two screws at top and bottom of amplifier plate.

DO NOT REMOVE SCREWS IN MIDDLE OF PLATE. THESE SCREWS ARE PERMANENT AND ARE NOT MEANT TO BE REMOVED.

2) Gently remove plate and amplifier from the housing by pulling plate straight out.

To plug in amplifier, perform the above steps in reverse order.

Control Panel Elements

(A) POWER INDICATOR: The correct power is being applied to the controller when the light is on.

(B) REFERENCE SWITCH: The three indicators display the status of the sensors, the reference switches of the axes. An axis is in "normal" position, i.e., off the sensor, when the indicators are OFF. When the system is first powered up, all three indicator lights must be OFF. If any light is ON, the mechanics should be adjusted. If there are less than 3 axes the appropriate DIP switch must be set.

(C) START: This momentary push button starts preprogrammed operations. No action takes place if there is no program in memory. In general, the first push of the button results in the system moving to the HOME position. A program can also be STARTED from the PC with the command @**0S**.

(D) STOP: This causes a program to stop. If a motion was taking place, a deceleration ramp is inserted in the program and the program execution is stopped. The interrupted program can be continued with the START button. A program can also be STOPPED from the PC with the command @**0d.**

(E) RESET: This resets the system as if the controller has been turned off and on again.

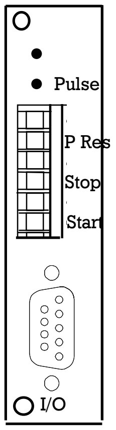
(F) EMERGENCY: This button immediately stops the program without ramping. This type of stop, Emergency Stop, usually causes the system to lose steps. Subsequently pushing the START button will result in a homing operation.

(G) DB9 RS232 INPUT: A 3-wire RS232 system is used to communicate with the system (transmit, receive, ground).

Hooking Up Impulse Control

☞ ***Utilizing impulse control is optional. If you do not wish to operate the controller using impulse signals, skip this section. Note: C108 Controllers only have impulse control capabilities if specially requested when ordered.***

If you wish to operate your controller through pulse signals sent from an external source, you can utilize the port on the rear of the controller (see page 6) designed for pulse start, stop, and reset signals. Also on this panel is a DB9 connector for auxiliary impulse inputs/outputs. A close up of the impulse panel is given below.



To hook up impulse signals:

1) Gently pull the connector with the screw terminals off the rear of the controller.

2) Loosen the screw on the side of the appropriate terminal for Start signal.

3) Insert one end of a wire into the space behind the screw.

4) Loosen the screw on the side of the terminal **above** the Start signal.

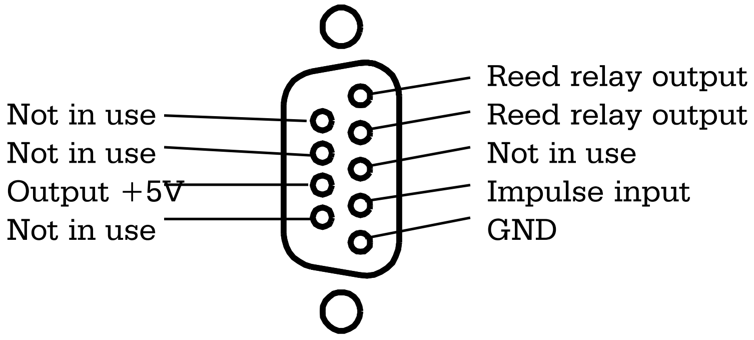
5) Insert an end of a second wire into the space behind the screw.

6) Tighten screws so that they hold each wire in place.

7) Repeat procedure for Stop and Reset (P Res) signals.

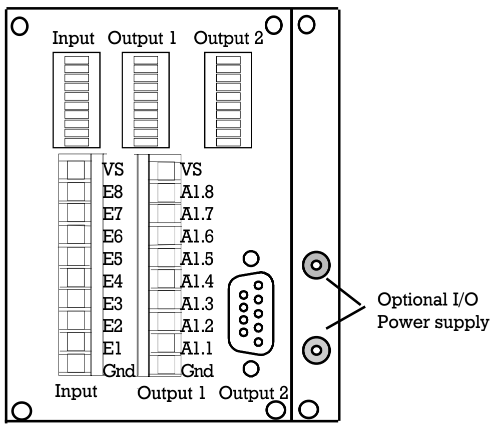
When the ends of the two wires are connected, the appropriate impulse signal will be activated.

The auxiliary input/output connector can be used to control input and outputs via pulse commands. A male DB9 connector should be attached to the female plug on the rear of the controller (see page 6 and page 10), with the other end of the wire attached to another driver card or device which will send/receive impulse signals. The pinout for the auxiliary I/O connector is given below:



Attaching Input/Output Devices

The input/output (I/O) connector ports are located on the center panel, on the rear of the controller (see page 6). A close-up of this panel is illustrated below:



The connector at the far left on the I/O panel is the INPUT PORT:

- The top screw terminal is V+ (the voltage supply connector).
- The bottom screw terminal is GND (ground)
- The screw terminals in-between, labelled E1 to E8 from bottom to top, are the connectors for inputs 1 to 8.

The center connector on the I/O panel is OUTPUT PORT1:

- The top screw terminal is V+ (the voltage supply connector).
- The bottom screw terminal is GND (ground)
- The screw terminals in-between, labelled A1.1 to A1.8 from bottom to top, are the connectors for outputs 1 to 8 of output group 1.

To attach input and output devices to either the INPUT PORT or OUTPUT PORT1:

(1) Gently pull the connector with the screw terminals off the rear of the controller.
(2) Loosen the screw on the side of the appropriate terminal for your input/output.
(3) Insert wire from input/output into the space behind the screw.
(4) Tighten screw so that it holds the wire in place.
(5) Plug the connector back onto the rear of the controller.

On the C-series Controllers, the Inputs and the Outputs on Group 1 are **optocoupled**. This means that there is complete electrical isolation between the controller and the input or output attached to the controller. Hence, you must supply power (approximately 50mA) from an outside source to the Input or Output Group 1 connectors (V+ and GND).

If you do not have an outside power source, you may use the optional power supply for optocoupled inputs/outputs, located to the right of the I/O panel (C108 controllers will not have this supply unless requested.) This is a 24V supply of up to 1A. To utilize this power source:

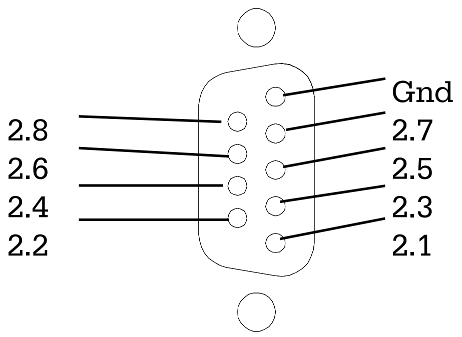
(1) Insert the banana plug end of a wire into the red (top) connector .

(2) Connect the other end of this wire to V+ terminal of the appropriate I/O port.

(3) Insert a banana plug end of a second wire into the blue (bottom) connector.

(4) Connect the other end of this wire to the GND terminal of the appropriate I/O port.

OUTPUT PORT2 is the female DB9 connector to the right of OUTPUT PORT1. The output assignments for OUTPUT PORT2 are illustrated on the following page.



Pin	Output
2.1	1
2.2	2
2.3	3
2.4	4
2.5	5
2.6	6

To attach outputs to OUTPUT PORT2 you need to use a cable with a DB9 connector. Plug the male end of the DB9 into OUTPUT PORT2 and connect the appropriate wire(s) at the other end of the cable to the appropriate outputs.

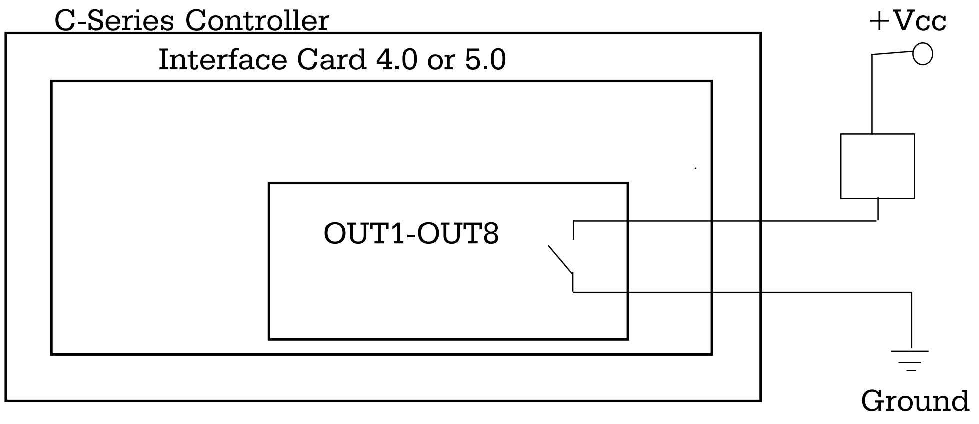
The outputs on Group 2 are not optocoupled. Hence these outputs are not isolated from the rest of the system and receive their power directly from the controller. These are TTL level outputs (+5V) and can supply 1.2 mA each. **Caution must be used when dealing with non-optocoupled outputs.**

The memory addresses for the input and output groups are as follows:

I/O	Memory address
Input	65531
Output Group1	65529
Output Group 2	65530

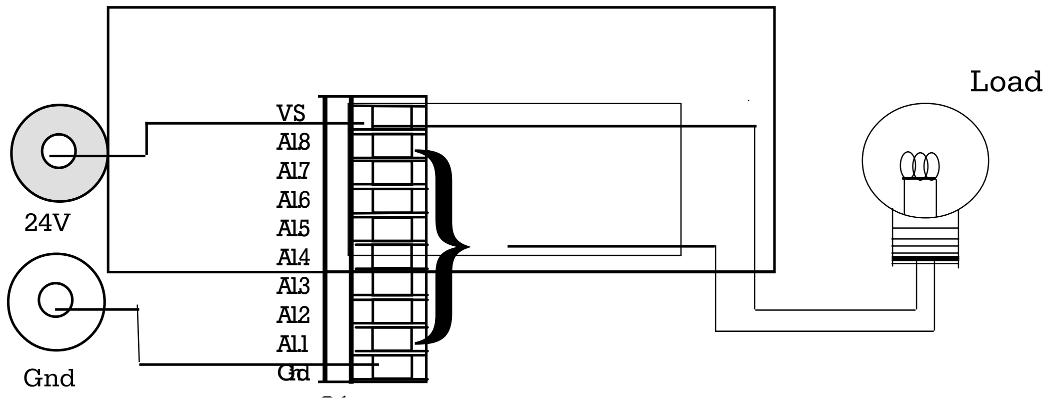
Connecting Output Devices

Output port 1 is optocoupled whereas Output port 2 is an "open collector" type output. In each case, the outputs act like switches to ground (Gnd) that complete the path for the load current. When you turn the output ON, the path is completed and current travels from the voltage source through the load, and then through the output switch to ground. When you turn the output OFF, the current path to ground is broken which then prevents current from flowing through the load. This is illustrated in the following diagram.



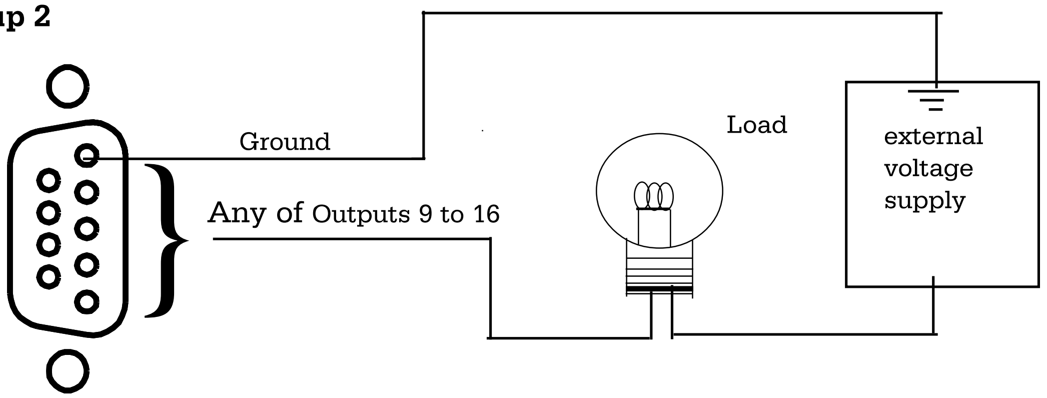
Connect one end of the load to the supply voltage connector (+Vcc) and the other end to one of the output connections, as illustrated on the following page.

Output Group 1



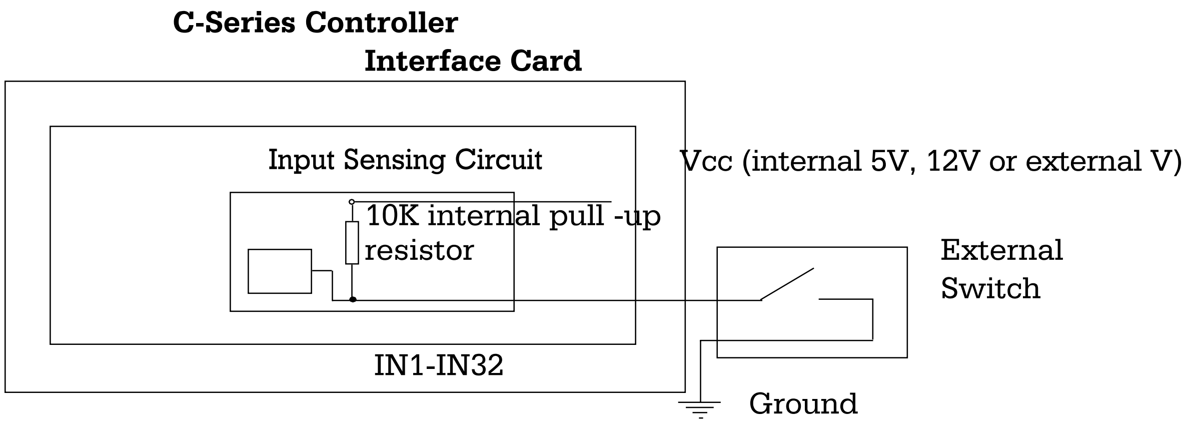
Note: the supply voltage (Vcc) can be either 24V from the optional power supply on the controller as shown above, or it can be an external supply voltage. **When using an external supply voltage, be sure that the Ground from the external supply is connected to the screw terminal marked Gnd.**

Output Group 2



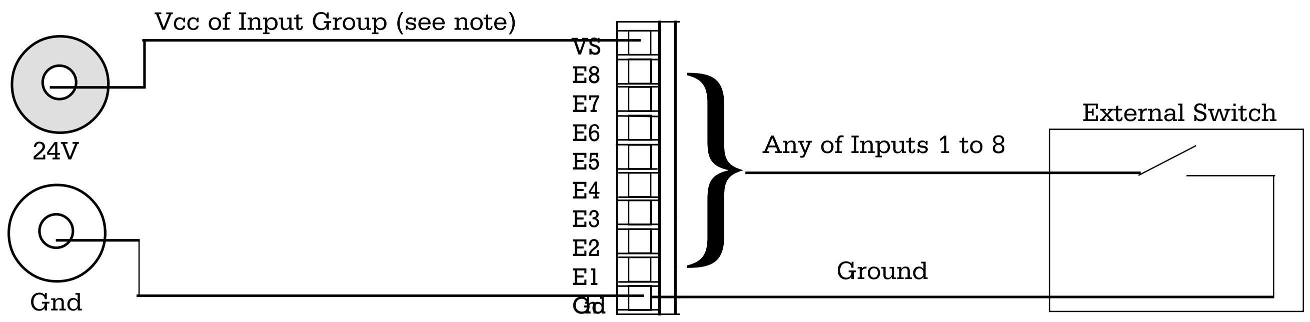
Connecting Input Devices

When an input switch is open or at no connection, the value read at the input is a logic high. The same holds true when the switch is at Vcc. This behavior is due to the internal pull-up resistor on the Interface card. When the switch contact is closed or connected to ground, the switch is read as logic zero. This is illustrated by the diagram below.



When you connect inputs, connect one end of the input switch to ground and the other end to the sense input line as indicated in the diagram below.

Input Group



Note: the supply voltage (Vcc) can be either 24V from the Interface card, or it can be an external supply voltage. **Be sure to hook up the Ground if you are using an external supply voltage.**

Attaching The Power Cable, Starting, And Using The Controller

(1) Connect the power cord by plugging it into the power connector on the far right of the rear of the controller. Plug the other end into a 110VAC outlet.

(2) Turn on your PC and wait for the DOS prompt:

- Insert the Intro to Isel disk into drive A.
- Enter the drive you wish to install the software on by typing the letter of the drive followed by a **:** Press ENTER.
- Make a directory called ISEL by typing **MD\ISEL** and pressing ENTER.
.
- Now enter that directory by typing **CD\ISEL** and pressing ENTER.
- You can now copy all the files from the disk by typing **COPY A:*.* /v** and pressing ENTER.
- Now type **COMTEST** at the DOS prompt and press to enter the communications program. From here you can run the self test and download programs from the PC to the controller.

(3) Turn on the controller by pressing the button to the left of the power connector (see page 6).

☞ ***If the controller does not work when you initially turn it on, release the red Emergency Stop button located in the upper left hand corner on the front of the controller by turning it clockwise. Sometimes the emergency stop is activated during shipping, and this will prevent the controller from turning on. If the controller still does not function, then examine section 4, Troubleshooting.***

(4) Run the Self-Test program, by pushing and holding down the START button for at least 1 second as you turn on the controller.

This program will cause the controller card to move the X and Y axes and send the ASCII character set over the serial line until it receives a character which you type on the PC. A minimum of 50 characters are sent. Once a character is received from the PC, all subsequently received characters are echoed back to the host.

☞ ***We have had difficulties with this program on various IBM-PC compatibles. The self-test program can also be run with a simple terminal or a terminal emulation program. The terminal should be set to 9600 baud, 8 data bits, 1 stop bit, no parity.***

If at any time you wish to exit the Self Test program type **X**. After the Self-Test program has been run, the controller must be reset by turning it off and then back on.

The Self-Test program is listed in section 5. Note that this program is given only for reference. **It is not necessary to type this program into your PC, as it has been preprogrammed and supplied on the Intro to Isel disk.**

(5) Finally, from this communications program, you can send both immediate and program commands directly to the controller. See section 3 for a description of these commands. See section 2 for a series of sample programs which you can use immediately.

Downloading Programs

Once you are back at the dumb terminal you can download programs to the controller. To do this, type `DOWNLOAD` to activate the download program. You will immediately be asked to give the name of the file to be downloaded. If you are downloading a file from another drive, you should indicate the drive and directory names.

This program transmits the file, line by line, to the controller card. The software acknowledge is checked after each line is transmitted. If an error occurs, the error message is displayed and the downloading is aborted.

The program automatically translates programs written on most word processors into the language understood by the controller. **However, programs must be prepared on an ASCII word processor that does not add any special formatting characters.**

This program is also listed in section 5. Note that it too is given only for reference. **It is not necessary to type this program into your PC, as it has been preprogrammed onto the Interface Card.**

Using A Memory Card With The Controller

Users can optionally purchase a Memory Card for use with the C Series of Controllers. The Memory Card provides permanent storage of motion control programs. You can then automatically download a program from the card to the controller without using a PC. Memory cards are available with memory capacities of 8K, 16K, and 32K.

To insert the Memory Card into the controller:

- Locate the Memory Card slot on the front panel of the controller (see page 10).
- Insert the Card into the slot so that the arrow on the Card points toward the controller. Note that the card will stick out about 1/2" when it is completely inserted.

To save a program onto the Memory Card.

- Insert the Memory Card into the Memory Card slot.
- Type a program into the controller’s memory in Program mode.

Move to Immediate mode by typing **9** (see page 29 for information on command modes).

- Once in Immediate mode, type the command @**0u**. The program which is in the memory of the controller will be copied onto the Memory Card.

SAVING A PROGRAM TO A MEMORY CARD WILL AUTOMATICALLY ERASE ANY PROGRAMS PREVIOUSLY STORED ON THE CARD. BE SURE THAT YOU DO NOT SAVE A PROGRAM ONTO A CARD CONTAINING A PROGRAM YOU WANT TO PRESERVE. THE MEMORY CARD CAN ONLY HOLD ONE PROGRAM, ALTHOUGH THAT PROGRAM MAY BE AS LARGE AS THE MEMORY CAPACITY OF THE CARD.

BE CAREFUL THAT YOUR PROGRAM IS NOT LARGER THAN THE MEMORY CAPACITY OF YOUR CARD. NO WARNING MESSAGE WILL APPEAR IF A FILE IS TOO LARGE FOR THE MEMORY CARD, AND ONLY AS MUCH AS WILL FIT ON THE CARD WILL BE COPIED TO IT.

You can now remove the Memory Card, and the program will be saved on the card. If you wish you can now run a program on the controller without first typing it in through the PC by using the Memory Card.

To download a saved program to the controller:

- Insert the Memory Card containing the desired program into the Memory Card slot.
- Press the reset button on the controller (see page 10). A copy of the program from the card will automatically be downloaded into the controller.

You can now run the downloaded program directly from the controller by pressing the Start button. Note that the PC is not involved in this procedure at all. Once a program is saved to a Memory card, the controller can be used to run the program independently of the PC.

2

SAMPLE PROGRAMS

This chapter contains a number of sample programs along with detailed explanations of their form and content. You may find these useful in a number of ways:

- You may wish to input and execute these programs immediately after running the tutorial as a way of learning to program by practice.
- You may wish to flip back and forth between the examples given here and the detailed explanations of individual commands given in chapter 3. This will give you a more in-depth knowledge of the commands before you actually start programming. You can then input some of these programs to gain hands-on experience.
- Finally, you may wish to actually use these programs or incorporate parts or versions of them into your own programs. Of course, this will depend on your own purposes, and these programs will only be worthwhile if they are appropriate.

The programs can be loaded into the controller with the download program.

WHEN RUNNING ANY OF THESE SAMPLE PROGRAMS, WE ADVISE YOU TO DO SO WITHOUT ANY SORT OF TOOL ATTACHED TO THE Z-AXIS AND WITHOUT ANYTHING MOUNTED OR SITTING ON THE TABLE.

Single Axis Motion With Initial Home

```
@01
@0d2000
@0i
71
0 1016,500
0 -406,300
9
@0S
```

- **@01**
  The first line of this program is in immediate mode. The command defines the X-axis (1) as the only axis to be moved in the following program.
- **@0d2000**
  This command is also in immediate mode. Here the homing speed is set to 2000 steps/second.
- **@0i**
  Although in immediate mode, this command indicates that programming is to shift into program mode. All following commands will be loaded into the controller but not executed until either an execute command is given or the start button is pressed.
- **71**
  This command homes the X axis.
- **0 1016,500**
  The fifth line of this program moves the X axis 1016 steps away from the motor at 500 steps/second.

- **0 -406,300**
  The sixth line of the program moves the X axis 406 steps toward the motor at 300 steps/second.
- **9**
  This line ends the program mode. All following commands will be in immediate mode and executed immediately.
- **@0S**
  This command signals that the controller should execute the program in memory. Hence lines five and six will be executed: the motor will home and then move the specified amounts.

Dual Axis Motion With Initial Home

```
@03
@0d2000,3000
@0i
71
72
0 1016,5000,0,2000
0 -406,300,1016,5000
9
@0S
```

- **@03**
  The first line of this program, in immediate mode, indicates that the X axis (1) and the Y axis (2) will be manipulated in the following motion (1+2=3).
- **@0d2000,3000**
  Here the homing speed for each axis is set. A homing speed of 2000 steps/second is set for the X axis and 3000 steps/second is set for the Y axis.
- **@0i**
  This command, in immediate mode, indicates that all following commands will be in program mode.
- **71**
  This command will home the X axis.
- **72**
  This command will home the Y axis. Note that the X and Y axes could have been homed with one command (73), but in that case, the Y would have been homed before the X.
- **0 1016,5000,0,2000**
  Line six gives motion commands for the two axes. The X axis is to move 1016 steps away from the motor at a velocity of 5000 steps/second. The Y axis is told to move 0 steps, so only the X axis moves in this line.
- **0 -406,300,1016,5000**
  Line seven also gives motion commands. Here the X axis is to move 406 steps towards the motor at a speed of 300 steps/second. Meanwhile, the Y axis is to move 1016 steps away from its motor at a speed of 5000 steps/second.

- **9**
  This command ends the program mode and shifts back to immediate mode.

- **@0S**
  The last line of the program sends a signal to the controller to execute the lines contained in memory. Hence lines four through seven will be executed at this time: both motors will home, X first and then Y, and then the motors will move as specified.

Three Axis Motion With Special Homes

This example illustrates a 3 axis system (X, Y, and Z). Note that the homing operation is performed in a specified order.

```
@07
@0d2000,3000,1000
@0i
75
72
0406,300,1016,5000,1016,3000,-1016,3000
0-406,300,-1016,5000,1016,3000,-1016,3000
9
@OS
```

- **@07**
  In immediate mode, this command indicates that all three axes, the X (1), the Y (2), and the Z (4) are to be moved in this program (1+2+4=7).

- **@0d2000,3000,1000**
  Also in immediate mode, line two sets the homing speed for each axis. The X axis is to home at 2000 steps/second; the Y axis is to home at 3000 steps/second; and the Z axis is to home at 1000 steps/second.

- **@0i**
  This immediate mode command indicates that all following commands are to be in program mode, that is, the commands will be stored in the controller until a start command is issued or until someone presses the start button.

- **75**
  This command indicates that the Z axis (4) and the X (1) axis should be homed (4+1=5). The Z axis will be homed before the X axis.

- **72**
  This line will home the Y axis. The Y axis is given a separate homing command because the user wants it to home after the X axis. If it was combined with the above command, the Y axis would be homed before the X axis.

- **0 406,300,1016,5000,1016,3000,-1016,3000**
  Line six contains motion commands for all three motors. The X axis is moved 406 steps away from its motor at a speed of 300 steps/second. The Y axis is moved 1016 steps away from its motor at a speed of 5000 steps/second. The last part of this line moves the Z axis twice. First the axis is moved down 1016 steps at a speed of 3000 steps/second. Then it is moved back up 1016 steps at the same speed.

- **0 -406,300,-1016,5000,1016,3000,-1016,3000**
  Line seven again moves all three motors. The X axis is moved 406 steps toward its motor at 300 steps/second. The Y axis is moved 1016 steps toward its motor at 5000 steps/second. The Z axis will move up and down in the same way it did above.

- **9**
  Line eight indicates the end of the program mode. All following commands will be in immediate mode.

- **@0S**
  Line nine sends a message to the controller to execute the program in its memory. Hence this line will execute lines 4 through 7 inputted above: the motors will home and then move as specified.

Repeated Motion And The Send Command

This example illustrates how the loop command can be used to create repeated motion and communication.

```
@07
@0d2000,3000, 1000
@Oi
71
74
72
01016,5000,0,2000,1016,3000,-1016,3000
0406,300,1016,5000,1016,3000,-1016,3000
0-406,300,-1016,5000,1016,3000,-1016,3000
175
260,0
550
36,-9
9
@0S
```

- **@07**
  In immediate mode, this command indicates that all three axes, the X (1), the Y (2) and the Z (4) are to be moved in this program (1+2+4=7).

- **@0d2000,3000,1000**
  Also in immediate mode, line two sets the homing speed for each axis. The X axis is to home at 2000 steps/second; the Y axis is to home at 3000 steps/second; and the Z axis is to home at 1000 steps/second.

- **@0i**
  This immediate mode command indicates that all following commands are to be in program mode, that is, the commands will be stored in the controller until a start command is issued or until someone presses the start button.

- **71**
  74
  72
  These three lines home the X, Z, and Y axes respectively. Notice that a separate home command is issued for each axis since the user wants the axes homed in this particular order. If all three were combined into one command, the order of homing would be Z, Y, and X.

- **01016,5000,0,2000,1016,3000,-1016,3000**
  This line indicates the motion for the three axes. The X axis is to move 1016 steps at 5000 steps/second. The Y axis is to move 0 steps. The last two pairs of numbers indicate that the Z axis should first move down 1016 steps at 3000 steps/second and then move up 1016 steps at 3000 steps/second.

- **0 406,300,1016,5000,1016,3000,-1016,3000**
  0 -406,300,-1016,5000,1016,3000,-1016,3000
  These two lines move the axes just as lines six and seven moved them in the last example.

- **175**
  This line sends the ASCII character 75 (the letter **K**) over the RS232 cable to the PC. This can be used as a signal to the PC, that the motion is completed.

- **260,0**
  This command halts the program until the user sends ASCII character 60 (the character <) back to the controller over the RS232 cable. After the user inputs **<**, execution will continue on to the next statement.

- **550**
  Line twelve activates a delay of about 5 seconds (.1 x 50). Thus after the user inputs 55, the program will still wait 5 seconds before continuing.

- **36,-9**
  This is the statement which activates the loop. The line indicates that the previous 9 lines (-9) should be repeated 6 more times. Hence, the axes will re-home and move, and the controller will send and wait for ASCII signals, six more times.

- **9**
  Line fourteen indicates the end of the program mode. All following commands will be in immediate mode.

- **@0S**
  Line fifteen sends a message to the controller to execute the program in its memory. Hence when inputted, this line will execute lines 4 through 13 inputted above.

Drilling A Pattern

The following program will drill a series of 2 rows each containing 6 holes, such as might be required for a 14 pin DIP socket. The holes will be on .1 centers and the rows will be separated by .3" as illustrated below (note: the following is based on a 4mm screw pitch, i.e. 2540 steps=1 unit):

```
Start of Row 1* ●  ●  ●  ●  ●  ●
                ●  ●  ●  ●  ●  ● * Start of Row 2
```

An algorithm to perform this function can be outlined as follows:

1) Go to the Starting Point of Row 1
2) Go .1" forward, drill a hole
3) Repeat (2) 5 more times
4) Go to the Starting Point of Row 2
5) go .1" backward, drill a hole
6) Repeat (5) five more times
7) Stop

```
@07
@0i
0 508, 9000, 508, 9000, 0, 9000, 0, 9000
0 254, 9000, 0, 9000, 2540, 1000,-2540, 9000
3 5,-1
0 254, 9000, 762, 9000, 0, 9000, 0, 9000
0-254, 9000, 0, 900, 2540,1000,-2540, 9000
3 5,-1
9
@0S
```

- **@07**
  In immediate mode, this command indicates that all three axes, the X (1), the Y (2) and the Z (4) are to be moved in this program (1+2+4=7).

- **@0i**
  This immediate mode command indicates that all following commands are to be in program mode, that is, the commands will be stored in the controller until a start command is issued or until someone presses the start button.

- **0 508, 9000, 508, 9000, 0, 9000, 0, 9000**
  This line moves the motors to the starting position for the drilling. The X axis and Y axes are each moved 508 steps at a speed of 9000 steps/second. Since the first number in each of the last two pairs is 0, the Z axis does not move at all; this is fitting since we are not actually drilling here, but moving into position.

- **0254, 9000, 0, 9000, 2540, 1000,-2540, 9000**
  This line makes one hole. The first pair of numbers moves the X axis 254 steps away from the motor, at a speed of 9000 steps/second, to the position of the first hole. The second pair indicates that the Y axis does not move, appropriate since we are moving across a row. The last two pairs of numbers move the Z axis and thus drill the hole. The first command moves the Z axis 2540 steps downward at a speed of 1000 steps/second, and the second command moves the Z axis the same amount upward at 9000 steps/second. Note the slower speed used for the downward motion when the drill is actually taking place.

- **35,-1**
  This line sets up a loop which will cause the last command (-1) to be repeated 5 more times. Hence the X axis will move over 254 steps, and the Z axis will drill a hole, both five more times, producing a total of six holes in the first row.

- **0 254,9000,762,9000,0,9000,0,9000**
  This line moves the motors to the starting position for drilling row 2. The X axis is moved 254 steps further, at 9000 steps/second, and the Y axis is moved 762 steps at a speed of 9000 steps/second. Since the first number in each of the last two pairs is 0, the Z axis does not move at all; this is fitting since we are again not drilling here, but moving into position.

- **0 -254,9000,0,9000,2540,9000,-2540,9000**
  This line drills one hole in the second row. The first pair of numbers moves the X axis 254 steps towards the motor, at a speed of 9000 steps/second, to the position of the first hole. The second pair indicates that the Y axis does not move, since we are again moving across a row. The last two pairs of numbers move the Z axis and thus drill the hole. The first Z axis motion is 2540 steps downward at a speed of 1000 steps/second, and the second Z axis motion is 2540 steps upward at 9000 steps/second. Note the slower speed used for the downward motion when the drilling is actually taking place.

- **35,-1**
  This line sets up a loop which will cause the last command (-1) to be repeated 5 more times. Hence the X axis will move over 254 steps and the Z axis will drill a hole, both 5 more times, producing a total of 6 holes in the second row.
- **9**
  Line nine indicates the end of the program mode. All following commands will be in immediate mode.
- **@0S**
  Line ten sends a message to the controller to execute the program in its memory. Hence this line will execute lines 3 through 8 inputted above, and cause the drilling of the two rows of holes.

Obviously this is a very useful program. Therefore we list below two other versions of this same program.

If you have purchased PAL software, the program would appear as follows:

```
#units inch/10;
move 2(9000), 2(9000), 0(9000), 0(9000);
repeat
    move 1(9000), 0(9000), 10(1000),-10(9000);
    until 7;
move 1(9000), 3(9000), 0(9000), 0 (9000);
repeat
    move -1(9000), 0(9000), 10(1000), -10(9000);
    until 7;
stop.
#start;
```

The same drilling pattern can be produced with a BASIC program using the controller board codes. The following program consists of a series of PRINT statements in BASIC that load the Techno Controller Card with the appropriate program.

```
100 OPEN "COM1:9600,N,8,1,RS,CS,DS,CD" AS #1
110 PRINT#1, "@07":GOSUB 1000
120 PRINT#1, "@0i":GOSUB 1000
130 PRINT#1, "0 508,9000, 508, 9000, 0, 9000, 0, 9000": GOSUB 1000
140 PRINT#1, "0 254,9000, 0, 9000, 2540, 9000, -2540, 9000": GOSUB 1000
150 PRINT#1, "3 5 -1":GOSUB 1000
160 PRINT#1, "0 254,9000,762, 9000, 0, 9000, 0, 9000": GOSUB 1000
170 PRINT#1, "0 -254,9000, 0, 9000,2540, 1000, -2540, 9000": GOSUB 1000
180 PRINT#1, “3 5 -1”:GOSUB 1000
190 PRINT#1, “9”:GOSUB 1000
200 PRINT#1, “@OS”:GOSUB 1000
210 STOP
 1000 IF LOC (1) > 1 THEN GOTO 1000
1010 A$=INPUT$ (1,1):IF A$=”O” THEN RETURN
1020 PRINT "Controller Card Error: “;A$
1030 STOP
```

3

PROGRAMMING THE CONTROLLER

The first sections of this chapter describe some important features of the communications system between the controller and your PC. This is followed by a detailed description of all the commands used in programming.

Command Modes

There are two modes in which you can issue commands for the controller.

Immediate Mode

The controller board is capable of processing single line commands directly. A command that is to be immediately executed always begins with **@**. For example:

@0A400, 9000	(Immediate X axis motion)
@0A300, 1200, 400, 5000	(Immediate XY axis motion)
@0A300, 1400, 340, 5500, 20, 90, 900, 100	(Immediate XYZ motion)

The controller sends back the acknowledge signal **0** after the command:

- If the command begins with a capital letter, the controller sends the acknowledge signal at the completion of the command.
- If the command begins with a lower case letter, the controller sends the acknowledge signal at the beginning of the command.

Program Mode

In Program mode, a sequence of commands is loaded into memory. These will not be executed until the controller receives a START command in immediate mode, or until you press the START button on the front panel. The program remains in memory until a new program is downloaded. If the controller is equipped with battery backed memory, the program will remain in the controller even after it is turned off.

☞ ***Some commands can only be used in Program mode, while others can only be executed in Immediate mode. Other commands can be used in either mode. See the command descriptions (pages 31-82) to find out in which mode(s) a command is valid.***

Software Handshaking

A software handshake is used to facilitate communications. This software handshake is necessary since only 3 lines (Transmit, Receive and GND) are used in the communications. The software handshake requires the following protocol:

- All transmissions from the PC to the controller must end with a carriage return {chr (13) } and may be optionally followed by a line feed {chr (10)}.
- The controller board acknowledges the transmission with 0 {chr (48) } if the message is accepted, or with an error message containing a

character not equal to 0 if there is a problem. The ASCII value of the character is the number of the error message. The complete table of error messages is shown in the Error Message section in Section 5. Individual error numbers are explained under appropriate commands.

- You must wait for the acknowledge signal from the controller before transmitting any additional information.

Converting Units For Motion Commands

The Techno Controller Card is programmed in steps, not in units of length such as millimeters or inches. The conversion from distance units to steps is done by either the PAL-PC software, which provides automatic scaling, or by the user. The conversion depends upon the number of steps per revolution of the motor, and the size of the screw pitch you are using. User units such as inches or millimeters can be converted to steps by using the following formula:

$$\frac{\text{400 steps}}{\text{1 Revolution}} \times \frac{\text{1 Revolution}}{\text{screw pitch (mm)}} \times \frac{\text{Number of mm in the user's unit}}{\text{User's unit of movement}}$$

To illustrate the use of this formula, let's figure out the number of steps in 1 inch for a 4 mm screw pitch.

By plugging in the desired unit (1 inch) and the size of the screw pitch (4 mm), along with the number of millimeters in an inch (25.4), the formula looks as follows:

$$\frac{\text{400 steps}}{\text{1 Revolution}} \times \frac{\text{1 Revolution}}{\text{4 mm}} \times \frac{\text{25.4 mm}}{\text{1 inch}}$$

When this calculation is done, the Revolution terms drop out, as do the mm units. The result is:

$$\frac{\text{400 steps x 25.4}}{\text{4 x 1 inch}} = \frac{\text{2540 steps}}{\text{inch}} = \text{2540 steps/inch}$$

You would multiply the number of inches in your desired movement by 2540 to get the number of steps in your motion. It is this number of steps which you would use for the distance parameter in motion commands.

For your convenience, here are the number of steps in the most commonly used units and screw pitches:

Unit	**Screw Pitch (mm.)**	**Steps per unit**
1 inch	2	5080
1 inch	4	2540
1 inch	5	2032
1 inch	25.4	400
1 mm	2	200
1 mm	4	100
1 mm	5	80

Command Descriptions

The following pages contain the commands or instructions used to program the Isel C series controllers. The commands are discussed alphabetically. At the end of this section is a list of all the commands, summarizing their formats and functions.

Each command is explained using the following format :

Name of Command

☐ Definition

Gives a quick one line description of what the command does and indicates whether it is used in Immediate mode, Program mode, or both.

☐ Format

Gives the syntax of the command, demonstrating the way it is actually inputted. If applicable, both the version of the command used in the Immediate Mode and the version used in the Program Mode are given. Often variables will be shown here with the command. A variable is usually placed within parentheses to distinguish it from an alpha character which would actually be used. The parentheses should **not** be used within the actual programs. These variables will stand for data, variables, etc., which are used in conjunction with the command. Explanations of these variables will also appear in this section.

☐ Explanation

A detailed explanation of the use and function of the command along with the options available in connection with the command.

☐ Example

An example of a line or lines containing the command along with an explanation of how the command functions in this particular instance.

☐ Possible Error Messages

Lists the error messages which may appear in conjunction with this command, along with explanations.

☐ See Also

A cross-reference guide indicating related commands, if any, which are used with the given command or which might assist in understanding the command.

Absolute Motion

☐ Definition

This command immediately moves the defined axes to the specified absolute location relative to a previously set home position. This command can be used in either Immediate or Program mode.

☐ Format

Immediate Mode:
@ (GN) M (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)
or
@ (GN) m (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

Program Mode:
M (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)
or
m (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

M or m indicates when the controller should send back an acknowledge signal. The controller sends back the acknowledge at the end of the motion with the **M** command and at the beginning of the motion with the **m** command.

(SX),(SY),(SZ1),(SZ2) stand for the speed of motion in steps/second for each axis. Each axis must be set separately. Speeds must be a values between 30 and 10000.

(SZ1) is the speed of the Z axis on the downward path while **(SZ2)** is the speed of the Z axis on the return path.

(GX),(GY),(GZ1),(GZ2) **stands for the displacement relative to the origin in steps for each axis. In actuality, this is the position you wish the axis to move to. Each axis must be set separately. The position must be in the range -8,000,000 to +8,000,000 steps.**

(GZ1) is the position of the Z axis on the first motion, and
(GZ2) is the position of the Z axis on the second motion.

□ Explanation

This command moves the axes in absolute motion. This means that the distance you specify for each motor is the number of steps from a previously set home or origin (this is set with the Zero Absolute Position command). In actuality, you are specifying the coordinates of the new position to which you wish the axes to move.

You must include a value for every axis, even if you do not wish an axis to move. In this case, specify the current position of the axis, so that it will stay at that position.

The command can also be used for one and two axis systems.

□ Example

```
@07
@0R7
@0M2700,3000,2000,2000,350,1000,0,1000
@0m2700,9000,1000,2000,350,1000,0,1000
```

The first of this series of commands indicates that all three axes are to be used. The second command then sets the current position of the axes as the home position. All absolute motion commands will thus be relative to this point.

Thus in the third command, the X axis will move 2700 steps away from the motor from this position at a speed of 3000 steps/second, and the Y axis will move 2000 steps away from the motor from that position at a speed of 2000 steps/second. The third pair of numbers indicates the downward motion of the Z axis, 350 steps at 1000 steps/second. The last pair of values indicates the upward motion for the Z axis. The 0 distance does not mean that the axis will not move, but that it will move to the origin, hence it will return to the position it was at before the downward motion. Since a capital M was placed at the beginning of this line, the controller will send an acknowledge to the PC after this motion is completed. The coordinates of the axes at the end of the motion will be (2700,2000,0).

The fourth command line also moves the axes absolutely. Since a lower case m starts the line, an acknowledge will be sent from the controller to the PC before the motion begins. The distance value given for the X axis is 2700. Since this is the position that the X axis is already at, the X axis will not move. The speed of 9000 steps/second is ignored, although it must be included. The distance given for the Y axis is 1000. Thus the Y axis will move to a position 1000 steps away from the origin set in line 1. In actuality, the Y axis then moves 1000 units toward the motor from its previous position of 2000. The Z axis will move just as it did in the last command.

☐ Possible Error Messages

2	Home switch is ON. The requested motion lies outside the reach of one of the axes. This axis must be manually moved from its current position and the program restarted.
4	Axis not specified
5	Illegal number of steps or an illegal sign has been added.
7	The number of parameters does not correspond to the number of axes that has been specified. The controller board requires parameters for all defined axes, even if the motion on an axis is for 0 steps.
D	Illegal speed has been specified outside the range 20 -10,000.

☐ See Also

Zero Absolute Position

Circular Interpolation

The circular interpolation command only works with interface card 5.0, which is in the C-142 controller.

☐ Definition

This command allows you to perform circular motions using the X and Y axes. This command can be used in both Immediate and Program modes.

☐ Format

Immediate Mode:
@ (GN) Y (B),(V),(E),(X),(Y),(R_X),(R_Y)

Program Mode:
Y (B),(V),(E),(X),(Y),(R_X),(R_Y)

@	signals that this command is in immediate mode.
(GN)	stands for the device number. The standard is 0.
Y	is the standard symbol indicating a circular interpolation command
B	Length of the arc in steps (circumference). How to compute this number is described after the example for this command.
V	speed of motion in steps/second. The speed must be a value between 30 and 10000.
X,Y	are the starting positions of the X and Y axes relative to the center of the arc.
R_X,R_Y	are values indicating the quadrant containing the starting point of the arc as indicated in the chart below:

	For Counterclockwise motion					For Clockwise motion				
Starting Quadrant	1	2	3	4			1	2	3	4
Value for R_X	-1	-1	1	1			1	1	-1	-1
Value for R_Y	1	-1	-1	1			-1	1	1	-1

E stands for the error factor. This number must be included to make sure that the stepper motor draws a smooth curve. The number is dependent on the starting point of the curve (X, Y), the radius (R), and the direction of travel. Compute the error by using the following formula:

$$\frac{R_X,R_Y\ [X(X-R_X)+Y(Y-R_Y)-R^2]}{2}$$

☐ Explanation

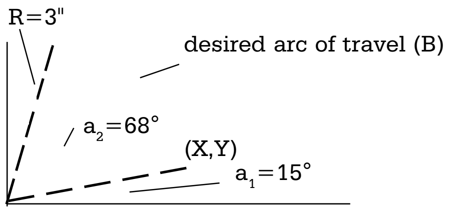
Before you use this circular motion command, you must first set the direction for circular motion using the SET CIRCULAR DIRECTION command. If you do not set this first, the controller will not know whether to perform clockwise or counterclockwise motion, and no motion will occur.

To perform the circular motion, you must input the starting point of the curve. This point is relative to the center point around which the arc is to be drawn. You must determine where you wish the center point to be, and hence the radius of the arc. However, the coordinates of the center point are not stated in this command or elsewhere. By computing the radius from the starting point you give, along with the distance of the arc, which you also input, the controller is able to create the correct size arc without you inputting the degrees in the curve.

☐ Example

```
@03
@0f-1
@0Y,9599,1000,314,7360,1972,-1,1
```

The first line of this example indicates that the x and y axes will be moved in the following commands. The second line sets all following circular motion to be performed counterclockwise. The next command gives the circular motion command that will draw the arc illustrated below.



The various parameters of this counterclockwise motion, assuming a 4mm pitch screw are figured as follows:

(X,Y) is the starting position of the arc, calculated by (R cos (a_1), R sin (a_1))
Therefore, (X,Y) = (7620[cos(15)],7620[sin(15)]) = 7360,1972).

The command opens with the machine number and the symbol for circular motion.

This is followed by the number of steps in the motion (B). To calculate the total number of steps in a circular arc motion (the circumference of the arc), one must calculate the total number of steps in each quadrant. (There are other ways, but they tend to be more complicated.)

In the diagram on the preceding page, the total X displacement is given by:

$DX = R \cos (a_1) - R \cos (a_2)$

and the total Y displacement is given by:

$DY = R \cos (a_2) - R \cos (a_1)$

The total number of points traveled is then given by the sum of these displacements in steps:

$B = DX + DY$

Consequently:

$DX = R \cos (a_1) - R \cos (a_2)$	$DY = R \sin (a_2) - R \sin (a_1)$
$DX = 7620 [\cos (15) - \cos (68)]$	$DY = 7620 [\sin (68) - \sin (15)]$
$DX = 4506$	$DY = 5093$

Total Number of Steps = DX + DY = 4506 + 5093 = 9599

The next value, 1000, gives the speed at which the motion will occur, 1000 steps/second. This is set arbitrarily by the user.

The next value, 314, is the error value which was computed using the error formula as follows (note that the values R_x,R_y used in this formula are calculated below.

$$\text{Error (E)} = \frac{R_x,R_y [X(X-R_x) + Y(Y-R_y)-R^2}{2}$$

$$= \frac{7620^2 - 7360 \times 7361 - 1972 \times 1971}{2}$$

$$= 314$$

The next two values give the coordinates of the starting point for the circle (as indicated in the above drawing) relative to the circle's center. As calculated in the note at the beginning of this example, (X,Y) = (7360,1972)

Finally, the values -1 and 1 are inputted as R_x and R_y as directed in the above chart for counterclockwise motion begining in the first quadrant.

☐See Also

Set Circle Direction

Clear Battery Backed RAM

☐ Definition

This command clears the memory of any program in the battery backed RAM. This command can only be used in Immediate mode.

☐ Format

```
@(GN) k
```

(GN) stands for the device number. The standard is 0.

☐ Explanation

Battery backed RAM is an optional feature of C-Controllers. With battery backed RAM, a downloaded program will remain in the controller even after the controller is turned off.

Loading a new program will automatically clear the memory of the battery backed RAM, but this command can be used to clear it without downloading a new program.

☐ Example

```
@0k
@07
@0 i
0 1000,2000,1000,2000,150,500,-150,500
9
@s
```

Here the user wishes to insert a program into the controller's memory, but beforehand he/she recognizes that there was already a program in the battery backed RAM. Hence the Clear Battery Backed Memory command is given before any other commands. Hence, the motion command in line four can be stored in memory, and the last command, a start execution command, will run the program without interference from that previous program. If the controller is now turned off, it is this program which will remain in battery backed RAM until another Clear Batter Backed RAM command is given.

End Program

☐ Definition

This command informs the controller board that no more program data will be transmitted in program mode, and thus the controller board is placed in Immediate mode. Obviously, this command can only be used in Program mode

☐ Format

```
9
```

☐ Explanation

This command is used to terminate transmission during Program mode. It is placed after the last command in Program mode and shifts control into Immediate mode. Once in Immediate mode, you can give an execute command and run the program just inputted.

If an error was encountered during program transmission, the controller board automatically switches out of Input mode and it is not necessary to send this command.

☐ Example

```
@07
@0 i
0 2700,3000,2000,2000,350,1000,-350,1000
9
@s
```

The first of this series of commands indicates that all three axes are to be used. The second command then shifts control into Program mode. Hence, the motion command in line three is not immediately executed, but stored in memory. Line four contains the End Input command. This indicates to the controller that there are no more commands which will be given in Program mode, and hence it shifts back into Immediate mode. Thus the last command, a start execution command, is given in Immediate mode syntax.

☐ See Also

Enter Program Mode

Enter Program Mode

☐ Definition

This command puts the controller board into Program mode so that all following data is stored and not executed until a Start command is encountered. Obviously, this command can only be used in Immediate mode

☐ Format

@(GN)i

(GN) stands for the device number. The standard is 0.

☐ Explanation

This command is used to terminate transmission in Immediate mode. It is placed after the last command in Immediate mode and shifts control into Program mode. After receiving the “i” command, the controller board waits for a string of data consisting of valid Program mode commands. These will be stored in memory and not executed immediately. The data must be closed with the End Program command.

This command must be preceded by the “Specify Axes” command.

The Enter Program Mode command clears the memory of all previous commands. If an error occurs during the following transmission of the data, the controller board terminates the Program mode automatically.

☐ Example

```
@07
@0 i
0 4500,4000,1030,4000,1967,1000,-1967,1000
9
```

The first of this series of commands indicates that all three axes are to be used. Line two then contains the Enter Program Mode command. The controller now awaits commands in Program mode syntax and stores these in memory. Hence, the motion command in line three is not immediately executed, but stored in memory. Line four closes the Program mode with the End Program command.

☐ Possible Error Messages

4	Axes not defined
5	Syntax error

☐ See Also

End Program

Home Motors

☐ Definition

This command sends the axes to their mechanical home positions. This command can be used in both Immediate and Program modes.

☐ Format

Immediate mode:
@ (GN) R (Axes) or **@ (GN) r (Axes)**

Program mode:
7 (Axes)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

R or **r** indicates when the controller should send back an acknowledge signal. The controller sends back the acknowledge at the end of the homing motion with the **R** command and at the beginning of the homing motion with the **r** command.

(Axes) Axis number. Each axis is defined by its own number: X = 1, Y = 2, Z = 4. If a combination of axes is to be homed, the axis number is the **sum of individual axis numbers.**

☐ Explanation

This command homes the axes at a speed previously set. This means that the axes indicated are sent to the mechanical home position, up against the home sensor for each motor.

When more than one axis is present, the Z axis is homed first, to clear the tool from the work area. The Y axis is then homed, followed by the X axis. If you wish the axes homed in a different order, you must issue a series of separate home commands.

In Immediate mode, additional commands cannot be sent until the mechanical system has been homed.

If the controller is told to home an axis with no mechanical component attached, the controller will continue to try and perform a homing operation and no acknowledge signal will be sent. **The STOP button must be pushed twice to exit from this state.**

☐ Example

```
@07
@0R7
```

The first of this series of commands indicates that all three axes are to be used. The second command is the home command as used in immediate mode. The 7 indicates that all three axes are to be homed (1+2+4). Thus the Z axis will be homed first, followed by the Y axis, and finally the X axis. Since a capital R was used, a signal will be sent from the controller to indicate the command was received after the homing is completed.

```
@07
@0i
7 5
9
@0S
@0r2
```

Here the axes are homed in a special order using both Immediate and Program modes. The first two lines indicate all axes may be moved and then move control into Program mode. The third line is a home command in Program mode. The 5 indicates that the Z and X axes are to be homed (1+4), but not the Y axis. This is done because the user wishes the Y axis homed after the X axis. A separate command must therefore be issued for the Y axis, since if it was included, it would automatically be homed before the X.

Line four moves into Immediate mode and the execute command in line 5 will cause the Z axis to be homed, followed by the X axis.

The last command is a home command in Immediate mode. Since a lower case **r** is given, a signal will be sent to the PC from the controller before the home is executed. Then the Y axis will be homed alone, as indicated by the 2.

☐ Possible Error Messages

3	The axes were defined, but a nonexisting axis was requested to be homed
4	Axes not defined

☐ See Also

Set Homing Speed
Specify Axes

Identify Axes Positions

◻ Definition

This command will display the current position of each axis This command can only be used in Immediate mode

◻ Format

@(GN)P

(GN) stands for the device number. The standard is 0.

◻Explanation

This command will display, on the communication screen the current absolute position of the axes in steps relative to the machine reference points (the home positions).

This positions will be displayed as a series of eighteen characters giving the absolute position using hexadecimal notation in 2's complement (See Section 5 for an explanation of hexadecimal notation).

◻Example

```
@07
@0 i
7 7
016,4000,8192,4000,0,1000,-2,1000
9
@0S
@0P
```

The first of this series of commands indicates that all three axes are to be used. Line two then enters program mode. Line three then gives the program command to home all three axes. Hence the motion command in line four will move the axes to position (16,8192,-2) relative to the reference positions. Line five then ends the Program mode and line six runs the program.

Line seven now shows the Indicate Axes' Position command. When sent to the controller with the axes in this position, the controller will respond with the following line to the Communication Screen:

```
000010002000FFFFFE
```

This indicates that the position of the X axis is 000010H or 16 steps from its home position, the Y axis is 002000H or 8096 steps from its home position, and the Z axis is FFFFFE or -2 steps from its home position.

Impulse Operation

☐ Definition

This command provides programmable control of the optional impulse input/output (I/O) port (see page 12). If the I/O port is not installed, this command can be used as a pause in the program. The command only works in Program mode.

☐ Format

4 (code)

(code) - any number between 1 and 6 as follows:

1 - output ON
2 - output OFF
3 - Impulse for .5 sec
4 - wait for impulse
5 - put out an impulse and wait for an acknowledgment. Retry after .5 sec.
6 - wait for an impulse and acknowledge

☐ Explanation

The Impulse Operation command is designed to interface the control card to other devices. The individual options allow for easy coordination with either other controllers or devices such as pneumatics etc.

Codes 1 and 2 are used to turn on or off the impulse output. Thus these commands can be used to turn on or off a device which has been connected to the impulse connector. Note: if a device is turned on, it will remain on through all future commands until it is turned off with a future command.

Code 3 sends impulses along the serial cable attached to the impulse I/O for 1/2 a second.

Code 4 causes the program to pause until an impulse is received from another controller card along the serial cable attached to the impulse I/O.

Code 5 sends an impulse signal to another controller card along the serial cable connecting the two, and then causes the program to pause until an acknowledge signal is sent in response. If no acknowledge is received, the controller will send another signal after .5 seconds.

Code 6 serves the same purpose, but instructs the controller to send an acknowledge signal to the other controller card once the impulse is received.

Note that since the impulse input is the same as the START input, when the output is ON, input cannot be recognized and Codes 4 to 6 cannot be used.

☐ Example

```
0 1500,2000,1500,1000,0,100,0,100
4 1
0 -1000,2000,1500,1000,375,100,-375,100
4 2
0 500,2000,1500,1000,0,100,0,100
4 5
7 7
```

This series of commands shows the impulse command used in a number of ways. (Note that all these commands are in Program mode and would not actually be executed unless the START button was pressed or Program mode was ended and a Start Command issued.)

The first line moves the axes into the desired position. Line two then uses the impulse command (code 1) to turn on the output (and hence a device connected to the output). The next line again moves the axes, this time with the device working.

The next line uses the impulse command to turn off the output (code 2) and hence the device. Thus the axes can be moved in line five without the device working.

Line six now uses the code to send an impulse signal to another controller card. Since code 5 was used to do this, the program will stop execution until an acknowledge signal is received from the other controller. Note that the output was turned off before sending this code since inputs could not be received if the output was on,. Once the acknowledge signal is received, execution will move to the next line and the axes will be homed. If the acknowledge signal is not received, another impulse will be sent out after .5 seconds.

☐ Possible Error Messages

5	Syntax error
6	Out of memory
7	Illegal parameters

☐ See Also

Move Until Impulse

Incremental Motion

☐ Definition

This command moves the defined axes (in the currently active interpolation mode) the specified distances relative to their previous positions. This command can be used in both Immediate and Program modes.

☐ Format

Immediate mode:

@ (GN) A (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

or

@ (GN) a (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

Program mode:

0 (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

A or **a** indicates when the controller should send back an acknowledge signal, The controller sends back the acknowledge at the end of the motion with the **A** command and at the beginning of the motion with the **a** command.

(GX), (GY), (GZ1), (GZ2)	stands for the displacement in steps for each axis. This is the number of steps you wish the axis to move from its last position. Each axis must be set separately. The position must be in the range 0 to ±8,000,000 steps.

(GZ1) is the distance the Z axis moves on the first motion while

(GZ2) is the distance the Z axis moves on the second motion path.

(SX),(SY),(SZ1),(SZ2)	stands for the speed of motion in steps/second for each axis. Each axis must be set separately. The speed must be a value between 30 and 10000.

(SZ1) is the speed of the Z axis on the downward path while **(SZ2)** is the speed of the Z axis on the return path.

☐ Explanation

This command moves the axes in incremental motion. This means that the distance you specify for each motor is the number of steps the axis will move from its previous position.

Positive values used for distance will move the axis away from its motor. Negative values used for distance will move the axis towards its motor.

Normally, the motion along the X-Y axis is interpolated and will be performed first. The motion along the Z1 direction will then be performed, followed lastly by the motion along the Z2 direction.

You must include a value for every axis, even if you do not wish an axis to move. If this is the case, insert a 0 for that axis' position so that it will not move any steps.

In Program mode, the controller sends back an acknowledge signal after the command is stored. In Immediate mode the signal is sent back before or after the motion depending on the case of the A/a.

If the controller is in 3 axis linear interpolation mode, the final Z axis displacement and speed should be set at 0 and 30 respectively.

This command can also be used for one and two axis motions.

MOTIONS ARE NOT CHECKED TO SEE IF THEY ARE WITHIN THE PHYSICAL RANGE OF THE MACHINE. MAKE SURE TO CAREFULLY CHECK THE DISTANCE YOU INSTRUCT THE AXES TO MOVE SINCE MOTION BEYOND THE AXES' RANGE CAN CAUSE DAMAGE TO THE MACHINERY.

☐ Example

```
@07
@0i
0 1500,2000,1500,1000,375,1000,-375,1000
9
@0s
@0A-2700,3000,0,2000,350,1000,-350,1000
@0a-2700,3000,0,2000,350,1000,-350,1000
```

This series of commands moves the axes three times using both Program and Immediate modes.

The first command indicates that all three axes are to be used. The second line then moves control from Immediate to Program mode.

The third command is an incremental motion command in Program mode. This line sets the X axis to move 1500 steps from its previous position away from the motor, at a speed of 2000 steps/second. The Y axis is set to move simultaneously 1000 steps from its previous position away from its motor at 1000 steps/second. The Z axis will first move 375 steps downward at 1000 steps/second, and finally move 375 steps upward at the same speed. None of these motions will actually occur; this line is stored in memory until an execute command is encountered. An acknowledge signal is immediately sent.

Line four ends the Program mode, and line 5, in Immediate mode, sends the signal to execute the program in memory. Hence at this point the motion from line three will be enacted. The X and Y axes will move first, in interpolated motion, and then the Z axis will move down and up,

Line six gives a motion command in Immediate mode. The X axis will move 2700 steps toward the motor from its current position at a speed of 3000 steps/second. The Y axis will not move since 0 was inputted for its distance. The speed assigned to Y is ignored. The third pair of numbers indicates the downward motion of the Z axis, 350 steps at 1000 steps/second. The last pair indicates the upward motion for the Z axis, 350 steps at the same speed. Since an upper case **A** was placed at the beginning of this line, the controller will send an acknowledge to the PC after this motion.

Line seven moves the axes exactly as in line six. However, since a lower case **a** starts the line, an acknowledge will be sent from the controller to the PC before the motion begins.

☐ Possible Error Messages

2	Home switch is ON. The requested motion lies outside the reach of one of the axes. This axis must be manually moved from its current position and the program restarted.
4	Axes not specified
5	Illegal number of steps or a syntax error.
6	Out of memory space
7	The number of parameters does not correspond to the number of axes that has been specified. The controller board requires parameters for all defined axes, even if the motion on an axis is for 0 steps.
D	Illegal speed has been specified outside the range 20 -10,000

☐ See Also

Absolute Motion

Loop/Branch

☐ Definition

This command can either cause a set of commands to repeat a set number of times (loop) or send execution to a particular command (branch). This command can only be used in Program mode.

☐ Format

3 (N), (S)

(N) Number of repetitions required for loop. Any number between 1 and 32767 may be used. If 0 is used, this becomes an unconditional branch.

(S) When used as a loop, this parameter indicates how far back from the current command the loop should begin. This must be a value between -1 and -32767. When used as a branch, this parameter is the instruction you wish to branch to, relative to the current command. This must be a value between -32767 and 32767.

☐ Explanation

When used as a loop (N>0), this command will send execution back to the command S lines back from this line. Execution will thus begin again from that line, and the commands between these two lines will be executed again. When execution again hits this line, the number of times execution has been sent back is compared to the value of N. If N is greater, execution will again be sent back and the lines re-executed. When the loop has occurred N times, execution will pass through this line and move to the next command.

- Forward loops (S>0) are not allowed.
- You can nest loops (place a loop within a loop), but nested loops are limited to a maximum depth of 4.
- At least one command must be included in the loop (S cannot equal 0).

When used as an unconditional branch, (N=0), this line will automatically send execution to the line which is S lines away from the current line. Execution will not repeat or loop back when the command is used as a branch.

- Forward branches (S>0) **are** allowed as well as reverse branches (S<0).
- Do not branch outside the defined program area. Unforeseen and unwanted results can occur, often resulting in damage to the machinery.

This command results in an internal counter being set up. If a Wait for Sync command causes branching outside of the loop, the counter is reset and execution is started from the instruction to which it was branched.

☐ Examples

```
0 1300,500
7 1
3,5,-2
```

The first of this series of commands, taken from the middle of a larger series of commands, moves the X axis 1300 steps at 500 steps/second. The next line homes the X axis.

The third line is the loop/branch command. It is immediately identifiable as a loop since the middle term is greater than 0. The last digit (-2) indicates that this loop should begin -2 lines, or 2 lines back from this line, that is, at the first motion command. The middle number, (5) indicates that the series of commands should be repeated five times. Hence when execution first hits here, the motion and homing (as seen above) will have occurred once. This command will send execution back 5 more times. Thus the X axis will be moved 1300 steps and then homed, a total of 6 times after these three lines are completed.

```
0 400,300,5000,300
7 2
3,4,-2
3,0,-3
```

The first line here moves the X axis 400 steps at 300 steps/second and the Y axis 5000 steps at the same speed. The second line then homes the Y axis.

The third line is a loop command. It indicates that the previous two lines (-2) should be repeated 4 times. Hence the axes will be moved as described above, four more times. After this, execution will pass through line three to the next line.

The fourth line is an unconditional branch, identifiable because the middle term is 0. Hence this line will automatically send execution to the line -3 lines away, that is, the first line. Here we see an example of a loop within a loop. When execution is sent back to line one, that loop will be re-executed as described above. Hence the axes will be moved another four times. Execution will then again pass through to this line which will again send it back to line one. This pattern will continue indefinitely, since line 4 is an unconditional branch, unless someone presses the STOP button on the controller.

☐ Possible Error Messages

5	Syntax error
6	Out of memory
7	Illegal parameters
C	An illegal loop or branch was encountered during execution.

Move Until Impulse

☐ Definition

This command moves the defined axes just as in the incremental motion command, but will terminate motion if an impulse is received at any time. This command can be used only in Program mode.

☐ Format

6 (GX), (SX), (GY), (SY), (GZ1), (SZ1), (GZ2), (SZ2)

(SX),(SY),(SZ1),(SZ2)	stands for the speed of motion in steps/second for each axis. Each axis must be set separately. The speed must be a value between 30 and 10000.
(SZ1)	is the speed of the Z axis on the downward path.
(SZ2)	is the speed of the Z axis on the return path.
(GX), (GY), (GZ1), (GZ2)	stands for the displacement in steps for each axis. This is the number of steps you wish the axis to move from its last position. Each axis must be set separately. The position must be in the range 0 to +8,000,000 steps.
(GZ1)	is the distance the Z axis moves on the downward path
(GZ2)	is the distance the Z axis moves on the return path.

☐ Explanation

This command moves the axes in incremental motion as described under the Incremental Motion command. However, when this command is used, an impulse input sent to the controller at any time during the motion will end the motion. Execution will then move on to the next command.

An impulse acknowledge is sent after execution of the command.

This command can also be used for one or two axis motions.

☐ Example

```
@07
@0i
6 1500,2000,1500,1000,375,1000,-375,1000
6 2700,3000,0,2000,350,1000,-350,1000
9
@0s
```

The first command in this short program, indicates that all three axes are to be used. The second line then moves control from Immediate to Program mode.

The third command is a Move Until Impulse command. This line sets the X axis to move 1500 steps from its previous position away from the motor, at a speed of 2000 steps/second. The Y axis is set to move 1000 steps from its previous position away from its motor at 1000 steps/second. The Z axis will first move 375 steps downward at 1000 steps/second, and then move 375 steps upward at the same speed. These motions will occur normally when this line is executed. However, if the controller receives an impulse input at any point, the motion(s) will immediately end and execution will pass to the next command. In any case, an impulse acknowledge is sent from the controller after the command is executed.

Line four gives another Move Until Impulse command. The X axis will move 2700 steps toward the motor from its current position at a speed of 3000 steps/second. The Y axis will not move since 0 was inputted for its distance. The speed assigned to Y is ignored. The third pair of numbers indicates the downward motion of the Z axis, 350 steps at 1000 steps/second. The last pair indicates the upward motion for the Z axis. 350 steps at the same speed. Once again, these motions will occur normally when this line is executed unless an impulse input is received by the controller. If an impulse is received, all motion stops, and execution passes to line 5. An impulse acknowledge is sent from the controller after the command is executed.

Line five returns execution to Immediate mode, and line six sends the signal to execute the program.

☐ Possible Error Messages

2	Home switch is ON. The requested motion lies outside the reach of one of the axes. This axis must be manually moved from its current position and the program restarted.
4	Axes not specified
5	Illegal number of steps or a syntax error.
6	Out of memory space
7	The number of parameters does not correspond to the number of axes that has been specified. The controller board requires parameters for all defined axes, even if the motion on an axis is for 0 steps.
D	Illegal speed has been specified outside the range 30 -10,000.

☐ See Also

Incremental Motion

Peek

◻ Definition

The Peek command allows the user to examine the RAM (read/write), as well as the EPROM via the serial interface.

◻ Format

@<GN>c<Addr> for Eprom
@<GN>b<Addr> for RAM

<GN>	=	Device number, standard = 0
<Addr>	=	Address between 0 and 65536

◻ Example

@0c2048
@0b4711

◻ Explanation

The card is addressed by **@0**. **c** means that it is required to read a value from the Eprom (the constant memory). **2048** is the address of the required value. The processor replies with the software handshake, followed by two digits, which represent the content of the required memory cell, expressed as a hexadecimal. If a value from the RAM is to be read, use **b** instead of **c** in the command code.

◻ Note

This command is intended to control the Input/Output Extension which is connected to the data/address bus of the interface card.

See page 93 for an explanation of hexadecimal numbers.

Poke

☐ Definition

The Poke command allows the contents of the RAM on the controller card to be modified via the serial port.

☐ Format

@<GN>B<Addr>,<Data>

<GN>	= device number
<Addr>	= address between 0 and 65535
<Data>	= value between 0 and 255

☐ Explanation

The card is addressed by **@0**. **B** means that a value is to be entered in the memory. **33211** provides the address for the value to be entered. **128** is the new value of this memory cell. The computer confirms execution with the software handshake.

☐ Example

@0B 33211,128

☐ Limitation

The command does not check whether the device connected to the data bus has in fact correctly accepted the data.

☐ Note

There is no command enabling the EPROM to be changed; nor would there be any point in this. The command is not intended to be used for altering internal card parameters, since the addresses of such parameters may change without warning. The command should not be used with addresses below 32767, because these addresses are used by the EPROM on the interface card.

Read Input Port

☐ Definition

This command sets up a program branch depending on the status of the input port. This command is only available in program mode.

☐ Format

o65531,(N),(V),(F)

65531	address of the input port
(N)	either stands for any of the bit numbers 1-8 corresponding to the appropriate input or 0 if the entire input address is being examined.
(V)	this is either 0 or 1 if a bit is being examined or a value between 0 and 255 if the entire address is being examined.
(F)	stands for an offset indicating the line to which branching should occur.

☐ Explanation

When execution reaches this line, either the entire input port address or a specific bit within that address (depending on the form of the command) is examined. If the value is equal to the value specified by **V**, then execution will branch to the line indicated by the offset. If not, then execution passes through to the next line.

If you want only a specific input checked, you should specify the bit number of the input in **N**, and then for **V**, indicate **0** if the input should be clear and **1** if the input should be set.

If you want all the inputs checked (i.e. the whole input port), you should place a zero for **N**. The value you insert for **V** should then be the decimal number whose binary value is the status of all the inputs (see the examples under the SET OUTPUTS command for more on binary/decimal values in addresses). Hence by using this command, you can cause a branch to take place only if a certain combination of inputs is set.

☐ Example

```
o 65531, 4,0,3
0 15, 2000
3 0 -2
7 1
9
```

The first line of this short program sets up the conditional branch. If input 4 is clear (i.e. bit 4 for the input port address 65531 is equal to 0), then execution branches to the line 3 lines ahead. This will send execution to line 4 which homes the axis, and then to line 5 which ends the program. If input 4 was set (i.e. bit 4 for input port address 65531 is equal to 1), then execution passes through line 1 to line 2 which moves the axis 15 steps, and then to line 3 sends execution back to the beginning of the program. Hence this program will keep checking the input 4 and moving the axis 15 steps as long as input 4 remains set. Once input 4 is clear, the axis will be homed and the program will end.

```
o 65531,0,9,3
m 15,2000
3 0 -2
7 1
9
```

The first line of this short program sets up another conditional branch, but this one will check the whole input port. According to line one, execution branches 3 lines ahead if input 1 and input 4 are both set and every other input is clear. This required status is indicated by the 9 since the binary value of 9 is 00001001, which means that the first and fourth bits are set (=1) and all the rest are clear (=0). If this is the case, execution will be sent to line 4 which homes the axis, and then to line 5 which ends the program. If any other pattern of inputs are set and/or clear, then execution passes through line 1 to line 2 which moves the axis 15 units, and then to line 3 returns execution to line 1. Thus this program will keep checking the input port and moving the axis until only inputs 1 and 4 are set.

☐ Possible Errors

5	syntax error
6	memory full
7	illegal parameters

☐ See Also

Set Outputs

Select Interpolation Axes

☐ Definition

This command selects which pair of axes will have interpolated motion and hence determines the plane of interpolation for linear or circular motion. This command can be used in Immediate mode or Program mode.

☐ Format

Immediate mode:

@ **(GN) e (S)**

Program mode:

e(S)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

(S) code to signal the axes the user wants interpolated, as follows:
0 sets interpolation on X-Y axes
1 sets interpolation on X-Z axes
2 sets interpolation on Y-Z axes

☐ Explanation

For all standard motions with the C-Series controller, the default is 2-D interpolation. This means that the motions programmed on two axes will occur simultaneously. The third axis will be moved after the other two, and needs a motion command for both forward and reverse motion.

This command thus determines which two axes will have interpolated motion. The non-interpolated axis is automatically assumed to require forward and reverse motion commands.

By default, the X and Y axes are set for interpolated motion, with the Z axis requiring a descending and an ascending command.

The form of interpolation set with this command will remain in effect until another command is sent, changing the interpolated axes.

The third axis motions need not be specified if the system only has two axes.

The command affects all motions, linear, circular, absolute, and incremental.

□ Example

```
@07
@0e0
@0A 2700,3000,1000,2000,350,1000,-350,1000
```

The first of this series of commands indicates that all three axes are to be used.

The second command sets the interpolation axes. Since 0 is used as the code value, motion on the X and Y axes will be interpolated. The Z axis will be moved after their motion and require two motion commands.

Thus in the motion command in line three, the X axis will move 2700 steps at 3000 Steps/seconds and the Y-axis will move 1000 steps at 2000 steps/second. These two axes will move first, in interpolated motion. The last two parts of the command are for the Z axis which will move forward (down) 350 steps at 1000 steps/second, and then move in reverse (up) 350 steps at 1000 steps/second, all after the XY motion.

```
@07
@0e2
@0A 2700,3000,1000,2000,350,1000,-350,1000
```

The first of this series of commands again indicates that all three axes are to be used.

The second command sets the interpolation axes. Since here 2 is used as the code value, motion on the Y and Z axes will be interpolated. The X axis will be moved after their motion and require two motion commands.

Thus in the motion command in line three, the Y axis will move 2700 steps at 3000 Steps/second and the Z axis will move 1000 steps at 2000 steps/second. These two axes will move first, in interpolated motion. The last two parts of the command are for the X axis which will move forward 350 steps at 1000 steps/second, and then move in reverse 350 steps at 1000 steps/second, all after the YZ motion.

□ See Also

3D Interpolation ON/OFF

Send Synchronization Character

◻ Definition

This command is designed to provide a synchronization signal for an external device, a second controller card or the host PC. When the Send Synchronization Command is encountered, the specified synchronization character is transmitted by the controller card over the serial line.

◻ Format

1 (SC)

(SC)	Synchronization number (ASCII code); any value between 33-125 except 64

◻ Explanation

This command is designed to coordinate the controller card with another controller card or with the host PC. The Send Synchronization command works in conjunction with the Wait for Synchronization command. The Send Synchronization command sends the designated character responding to the ASCII decimal equivalent over the serial line and will thus activate any other controller card that was waiting for that particular character.

The communication Self test program discussed on pages 16 and 17 can be used to test the Synchronization command.

A list of available ASCII codes is given in section 5.

◻ Example

1 84

When a program hits this line, the card will send the synchronization character T (ASCII character 84) over the serial line. A controller card on the other end, which has been told to wait for this character using the Wait command, will thus be activated at this point.

◻ Possible Errors

5	illegal synchronization character used (SC<33, SC>125, or SC=64)
6	memory full

◻ See Also

Wait for Synchronization Character

Set Circle Direction

☐ Definition

This command indicates the direction of circular motion for future circular motion commands. This command is in Immediate and Program Modes.

☞ ***The circular interpolation command only works with the C-142 controller that has an interface card 5.0***

☐ Format

Immediate mode:

@ (GN) f (D)

Program mode:

f(D)

@	signals that this command is in Immediate mode
(GN)	stands for the device number. The standard is 0.
f	is the standard symbol indicating the set circular direction command.
D	stands for the numerical value indicating clockwise or counterclockwise direction: use **-1** for counterclockwise direction or **0** for clockwise direction.

☐ Explanation

Before you can perform any type of circular motion, you must first indicate the direction of that motion (clockwise or counterclockwise) using this command in the immediate mode.

Once this command has been programmed, all following circular interpolation commands will be in this direction until a new Set Circle Direction command is used, changing the direction.

☐ Example

```
@03
@0f-1
@0Y,9599,1000,314,7360,1972,-1,1
@0f0
```

The first three commands here, all in Immediate mode, are used to draw the arc described on page 35 in the circular interpolation command description. Line two contains the Set Circle Direction command. Since the digit following the **f** is -1, this command indicates that all following motions, as in line three will be in a counterclockwise direction. Following the drawing of the circle, in line four, is another Set Circle Direction command. Since in this line, a 0 follows the f, all future circular motions will be run clockwise.

☐ See Also

Circular Interpolation

Set Homing Speed

☐ Definition

This command indicates the speed at which the axes will travel when they are homed. This command is only available in Immediate Mode.

☐ Format

@ (GN) d (SX), (SY), (SZ)

@ signals that this command is in immediate mode

(GN) stands for the device number. The standard is 0.

d is the standard symbol indicating the set homing speed command

(SX), (SY), (SZ) stand for homing speed in steps/sec. being assigned to each axis. These must be values between 30 and 10,000:

☐ Explanation

The default speed for the homing operation is set at 2000 steps/sec. The default can be overridden by using this command. The new value is preserved if the interface board in your controller has battery-backed memory.

You can set the homing speeds for one, all, or any combination of two axes. The speeds are always assigned to axes in X, Y, Z order; hence, the X axis always receives the first value, when present, the Y axis receives the next value, and the Z axis receives the last value. Which axes are assigned speeds depends on which axes were defined using the Specify Axes command, and the number of speeds must correspond to the number of axes defined.

A LARGE INDUCTIVE SPIKE IS PRODUCED WHEN THE MOTOR IS STOPPED ABRUPTLY AS IT HITS THE LIMIT SWITCH. THE INDUCTIVE SPIKE CAN BE LARGE ENOUGH TO OVERLOAD THE STEPPER DRIVER CARDS AND DAMAGE THE CONTROLLER. FOR THIS REASON, WE RECOMMEND THAT THE HOME SPEED BE KEPT TO UNDER 5000 STEPS/SEC.

☐ Examples

```
@07
@0d 3000,4000,250
```

The first line in this pair defines all three axes to be used in future commands. Thus in the second line, the Set Homing Speed command sets a speed of 3000 steps/sec. to the X axis, 4000 steps/sec. to the Y axis, and 250 steps/sec. to the Z axis. All future homing commands will now use these speeds.

```
@05
@0d 1000,250
```

The first line in this pair defines only the X and Z axes to be used in future commands. Thus in the second line, the Set Homing Speed command first sets a speed of 1000 steps/sec. to the X axis. Since the Y axis was not defined, the second value, 250 steps/sec., is assigned to the Z axis. The Y will thus remain at the default of 2000 steps/sec. All future commands will then use these speeds.

☐ Possible Error Messages

4	Axes not defined
5	Illegal number of steps or a syntax error.
7	The number of parameters does not correspond to the number of axes that has been specified.
D	Illegal speed has been specified outside the range 30 -10,000.

☐ See Also

Home Motors
Specify Axes

Set Outputs

□ Definition

This command allows you to turn on or off selected outputs. It is only available in Program mode.

□ Format

p (address),(N),(V)

(address) stands for the address of the output port you wish to adjust. This is either **65529** (Output Group 1) or **65530** (Output Group 2).

(N) either stands for any of the bit numbers 1-8 if a single output is to be changed, or 0 if multiple outputs are being adjusted at the address.

(V) this is either 0 or 1 if a bit is being adjusted or a value between 0 and 255 if the entire address is being adjusted.

□ Explanation

This statement can be used to turn either individual outputs or groups of outputs on or off.

There are two groups of eight outputs (bits 1 to 8) at each of the two addresses.

If you wish to adjust a single output, substitute the appropriate address for the group containing the output, and the bit number of the desired output for **N**. For the **V** parameter, substitute a **0** if you wish the output cleared (turned off) and a **1** if you wish the output set (turned on).

If you wish to adjust a number of outputs at once, first substitute the appropriate address for the group you wish to alter and then place a **0** for **N**. The number used for the **V** parameter should be a decimal number whose binary equivalent gives the status of each of the eight bits (hence outputs) in the address.

□ Example

```
p 65529,2,0
p 65530,6,1
p 65529,0,205
p 65530,0,50
```

This program shows a series of output adjustments.

The first line shows an individual output being turned off. Specifically, in Output Group 1 (address 65529) output 2 (N=2) is assigned a value of 0 and hence turned off.

The second line shows an individual output being turned on. Here, in Output Group 2 (address 65530), output 6 (N=6) is assigned a value of 1, and hence turned on.

Line 3 shows how an entire group can be adjusted. Here Output Group 1 is being adjusted (address 65529). The program knows the whole address is being adjusted since a 0 was used for **N**. The **V** then used to adjust the outputs is 205. This was determined by converting the binary number representing the desired status of all eight outputs into a decimal number as shown below:

Output	Desired Status	Binary Digit	Decimal Value
8	on	1	128
7	on	1	64
6	off	0	0
5	off	0	0
4	on	1	8
3	on	1	4
2	off	0	0
1	on	1	1
	Total:	11001101	205

The decimal value 205 thus stands for this particular series of outputs. Hence this line will turn on outputs 1,3,4,7, and 8, and turn off outputs 2,5, and 6 in Output Group 1.

Line 4 also shows how an entire group can be adjusted. Here Output Group 2 is being adjusted (address 65530). The program again knows the whole address is being adjusted since a 0 was used for **N**. The **V** then used to adjust the outputs is 32. This was determined by converting the binary number representing the desired status of all eight outputs into a decimal number as shown below:

Output	Desired Status	Binary Digit	Decimal Value
8	off	0	0
7	off	0	0
6	on	1	32
5	on	1	16
4	off	0	0
3	off	0	0
2	on	1	2
1	off	0	0
	Total:	00110010	50

The decimal value 50 thus stands for this particular series of outputs. Hence this line will turn off outputs 1,3,4,7, and 8, and turn on outputs 2,5, and 6 in Output Group 2.

☐ Possible Error Messages

5	Syntax error

Software Reset

□ Definition

This command tells the controller to reset the system. It is similar to pressing the RESET button of the controller. After entering this command, the controller can be warm started from the beginning of the program using the START command.

□ Format

ASCII character 254 (hexadecimal code is FEh)

□ Explanation

This command, given in Immediate mode, will halt the controller and reset it over the RS232 serial communications port.

If a motion was taking place when the command is received, no deceleration is used, and the motion is immediately halted.

□ Example

ASCII character 254

The Reset command, which would be sent by itself as shown above, will stop the program currently being executed by the controller.

□ See Also

Start Program
Stop Execution

Specify Axes

☐ Definition

This command defines the axes, indicating which axes will be used in all following commands. This command is only available in Immediate Mode.

☐ Format

@ (GN) (A)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

(A) stands for a number indicating which axes are to be defined. Each axis is known by its own number: X =1, Y=2, Z=4. If you wish to define a combination of axes, the axis number **A** is the **sum of the individual axis numbers.** The X axis must always be present since the power for the controller board is taken from the X axis power output

Hence, the valid axis definition numbers are:

1	x axis only
3	x and y axes only
5	x and z axes only
7	all three axes

Invalid definition numbers include:

0	no axis defined
2	y axis cannot be defined without x
4	z axis cannot be defined by itself
6	y and z axes cannot be defined without x
A>7	illegal number, not a valid combination of axes

☐ Explanation

This command must be sent to the controller before any other operations are performed. The Specify Axes command initializes the controller board for the connected axes, and the memory is cleared and partitioned for the defined axes.

☐ Examples

@07

The first line in this pair defines all three axes to be used in future commands because the number 7 was used to define the axes: 7= 1 (X axis) +2 (Y axis) + 4 (Z axis).

```
@03
```

In this case only the X and Y axes are being defined, since 3 = 1 (X axis) + 2 (Y axis). Any future commands which include the Z axis will thus cause an error message.

☐ Possible Error Messages

3	Illegal number of axes

Start Execution

☐ Definition

This command tells the controller to run the program which has been sent to memory using program mode. This command is only available in Immediate Mode.

☐ Format

@ (GN) S or **@(GN) s**

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

S or **s** indicates when the controller should send back an acknowledge signal, The controller sends back the acknowledge at the end of the program with the **S** command and at the beginning of the program with the **s** command.

☐ Explanation

Having already moved to Immediate mode, this command tells the controller to execute the commands previously sent to it in Program mode. Depending on which form of the command you use, the controller will send back an acknowledgment before execution or after executing all the commands in memory.

Note that you could also execute the commands in memory by pressing START on the controller. This command is useful, however, if you wish to start the program using the RS232 serial communications.

☐ Example

```
@07
@0i
7 7
0 1500,2000,1500,1000,375,1000,-375,1000
9
@0s
```

The first line in this pair defines all three axes to be used in future commands. The second line then moves from immediate mode into program mode.

Lines three and four are commands in Program mode. Line three will cause the axes to be homed, and line four moves the axes using incremental motion. These commands are not executed immediately, but stored to memory awaiting a start command or for the user to push the start button. Line five ends Program mode and moves into Immediate mode.

Line six gives the start command. Since a lower case s is used, the controller will send back an acknowledge signal immediately. Then the two programmed commands (lines three and four) will be executed. Hence at this point, the axes will be homed and then moved the desired amount.

Stop Execution

❑ Definition

This command tells the controller to abort the program which the controller is currently executing. This command is only available in Immediate Mode.

❑ Format

ASCII character 255 (hexadecimal code is FFh)

❑ Explanation

This command, given in Immediate mode, will halt a program which is currently being run by the controller. Giving this command is identical to pressing the STOP button on the controller but allows you to stop the program using RS232 serial communications.

If a motion was taking place when the command is received, a deceleration ramp is inserted into the program. The interrupted program can be continued by pressing the START button or with the Start Execution Command.

❑ Example

ASCII Character 255

The Stop Execution command, which would be sent by itself as shown above, will stop the program currently being executed by the controller.

❑ Notes

The Stop Execution command stops the execution of the current program. In case another command is in the buffer, the Stop Execution command should be entered a second time, otherwise the buffered command will start running.

❑ See Also

Start Program
Software Reset

Time Delay

☐ Definition

This command causes a program to pause for a preprogrammed time. It is only available in Program Mode.

☐ Format

5 (Time)

5 signals the time delay command

(Time) any number between 0 and 32767 representing the amount of time the program should pause. Each unit equals .1 sec.

☐ Explanation

This command causes a program to wait for the specified time before executing the next command. This can be useful if you need to make adjustments to the material or if motion must pause while outputs take effect.

☐ Example

```
0 2000,1000,1500,1000,200,-200
5 300
0 -1000,1000,-750,1000,200,0
```

The first line of this program moves the axes incrementally. The next line uses the Time Delay command to set up a delay of 30 seconds (.1 x300). In this case, the user might use this time to switch the materials on the table or to let an output take effect. After 30 seconds the next line will automatically be executed, thus moving the axes again incrementally to a new position.

NOTE THAT IT IS NOT RECOMMENDED THAT YOU USE DELAYS TO CHANGE MATERIAL BECAUSE INJURY MAY RESULT IF THE DELAY FINISHES AND MACHINING RESUMES BEFORE THE CHANGE IS MADE.

☐ Possible Error Messages

5	syntax error
7	illegal parameter

Trace

☐ Definition

The interface card carries out each stored command individually. After each command a signal is expected at the serial port, and the command status, including all relevant parameters, is given.

☐ Format

@<GN>t

@ signals that this command is in immediate mode
GN stands for the device number. The standard is 0.

☐ Explanation

The interface card carries out commands as usual, but before each command the command counter status is emitted as a simple integer. Then the command number and the operational constants with their appropriate data are emitted. The line is concluded with CR. The processor then waits for a signal at the port, and, following this, the command is carried out.

The function proceeds as follows for each to be run:
(A): The Trace-string is emitted (see below).
Wait for signal.
If signal 127 arrives, then mP-reset is triggered.
Command is executed
If command = "end of data block" then stop.
Otherwise next command after (A).

The Trace-string transmitted for each command has the following structure:

```
01234   00001   30 000001  FE87 ............ FFFF01   FE01
A           B         C     D         E                      F        G
```

(A) Memory address. This tells where the command is situated within the memory of the interface card.
(B) Command counter. This gives the number of the current NC command.
(C) NC command code. This gives the command to be executed. It is expressed in hexadecimal code and refers to the ASCII value of the command. In the above example, the command stored is "0"= "incremental move".
(D) X-axis displacement parameter. In the above example it is the 24-bit hexadecimal representation of the move in binary code.
(E) Value of speed for x-axis. To reconvert the speed, the fraction 921600/(HIbyte*(256-LObyte) can be formed.
(F) Command parameter as at (D), but for Z2.
(G) Speed parameter for Z2.

For other commands the parameters are disposed of in the correct order either as characters, or in binary code.

□ Limitation

Note 1:
For synchronizing commands, the first signal to be entered is the Trace function, followed by the synchronizing signal. "<Spacebar>", "<TAB>" or "<Linefeed>" cannot be used to continue single stepping.

Note 2:
To discontinue the Trace mode, you have to switch off the interface card, or, in response to the request for a signal, enter DEL (CHR[127]).

Note 3:
Due to technical developments, it is possible that the order of parameters and of storage may be different for various software versions, without prior notification.

Note 4:
The number of parameters transmitted back changes according to the axes selected, as follows:

x:	Command code +	6 byte
xy:	Command code +	10 byte
xz:	Command code +	15 byte
xyz:	Command code +	20 byte

Wait For Synchronization Character

□ Definition

Pause execution until synchronization character is received. This command can only be used in Program mode.

□ Format

2 (SC) (Off)

(SC) Synchronization number (ASCII code); any value between 33-125 except 64

(Off) Offset, any numerical value between -32767 and 32767.

□ Explanation

This command is designed to receive a synchronization signal from an external device, a second controller card, or the host PC. When the Wait for Synchronization Command is encountered, execution will pause until a character is received by the controller card over the serial line.

- If the ASCII decimal value of the character received is equal to the synchronization value **SC**, execution will pass to the next command in the program.
- If the ASCII decimal value of the character received is equal to the synchronization value **SC** +1, then execution will pass to the command to the instruction which is the number of lines away indicated by the offset.

The controller does not check to see that an offset will send execution to a valid instruction. **Note that branching out of the range of the actual program can produce unexpected results (and possibly dangerous if machinery is involved).**

The Wait for Synchronization signal command can only process one signal at a time. If synchronization signals are sent too quickly, they may be lost in transmission. It is advisable that acknowledge signals are sent to verify the receipt of a synchronization signal before a new signal is sent.

The communication test program discussed on page 16 and 17 can be used to test the synchronization command.

A list of available ASCII codes is given in section 5.

☐ Example

```
0 400,500,200,500,0,100,0,100
7 7
2 120 -2
1 50
0 1000,1000,2000,1000,150,100,-150,100
```

The first line of the series of commands moves the axes incrementally. The second line then homes all axes.

In the third line, the program hits the Wait command. Thus execution will pause until the synchronization character x (ASCII 120) is received from another controller.

- If the signal x is received, execution will pass to the next line.
- If the signal y (ASCII 121) is received, execution will pass to the line -2 lines away from this line, that is, the first line of this list. Hence all the previous lines will be repeated. This shows how the synchronization signals can be used to cause a loop.

The fourth line sends an acknowledgment synchronization signal to the other controller. Assumably the other controller is not waiting for this signal (via a Wait for Synchronization command) before it sends another. If this setup was not followed, and the other controller was just sending a stream of synchronization signals (e.g.. x,$,/,Z etc.), the signal x sent from the controller might be lost and the program never proceed.

Thus, after the synchronization character x has been received, execution will pass through line 4, send the acknowledgment and then pass to line five which moves the axes.

☐ Possible Errors

5	illegal synchronization character used (SC<33, SC>125, or SC=64)
B	illegal branching

☐ See Also

Send Synchronization Character

Zero Absolute Position

☐ Definition

This command sets the current position registers to zero, making the current position the origin. It is available in Immediate and Program modes.

☐ Format

Immediate mode:

@(GN) l,c,n (A)

Program mode:

n(A)

@ signals that this command is in immediate mode

n signals the zero absolute position command

(GN) stands for the device number. The standard is 0.

(A) stands for a number indicating which axes are to be set to zero. Each axis is known by its own number: X=1, Y=2, Z=4. If you wish to define a combination of axes, the axis number **A** is the **sum of the individual axis numbers.**

☐ Explanation

When using absolute motion, you specify the number of steps from a previously set home or origin for each axis. That origin is set using this command. The command sets the absolute position registers to 0 at the current position of the axes. All future absolute motion commands will thus be relative to this point.

The new origin set with Zero Absolute Position Command overrides any previous origin.

It is possible to set the registers on only one or two of the axes. In this case, the new origin will only apply to those axes, while the old origin will still apply to the other(s).

☐ Example

```
@03
@0n3
@0M2700,3000,2000,2000
@0M2700,9000,900,2000
@0n3
@0M2700,3000,2000,2000
```

The first of this series of commands indicates that only the X and Y axes are to be used.

The second command then uses the Zero Absolute Command to set the current position of the axes as origin. All absolute motion commands will thus be relative to this point.

Hence in the third command, the X axis will move 2700 steps away from the motor from this position at a speed of 3000 steps/second, and the Y axis will move 2000 steps away from the motor from that position at a speed of 2000 steps/second. The coordinates of the axes at the end of the motion will be (2700,2000), relative to the origin in line 2.

The next line, will also move the axes relative to the origin set in line 2. Hence , the X axis will not move at all, since it is already at position 2700 . The Y axis will move 1100 steps towards the motor from its current position, to the position 900 steps from the origin. The coordinates of the axes at the end of the motion will be (2700, 900), relative to the origin in line 2.

Line five now uses the Zero Absolute Position command to reset the origin. The current position, previously known as (2700,900) will now become (0,0). All future absolute motion commands will thus be relative to this new origin.

Hence in line six, which moves the motors just like line three, will nonetheless move them to a different overall position since they will be moved relative to this new origin. The coordinates after the motion will be (2700,2000) relative to the origin in line 5.

☐ Possible Error Messages

3	axes were not defined
5	syntax error

☐ See Also

Absolute Motion Command

3-Axes Interpolation On/Off

☐ Definition

This command places the controller alternately into 3-axes interpolation or standard 2-axes interpolation mode. This command is available both Immediate and Program modes.

☞ ***The 3-axes interpolation command only works with the C-142 controller, which has an interface card 5.0.***

☐ Format

Immediate mode:

@ (GN) z (I)

Program mode:

z (I)

@ signals that this command is in Immediate mode

(GN) stands for the device number. The standard is 0.

z indicates that the 3-Axes interpolation mode is to be set.

(I) stands for a number indicating whether the 3-AXES interpolation should be turned on or off. If a **0** is used, 3-AXES interpolation will be turned off. If a **1** is used, 3-AXES interpolation will be turned on.

☐ Explanation

Normally, the controller works in 2-AXES interpolation, that is, for all motion commands two axes will be moved in interpolated motion, and the third axis will be moved after their motion via two motion commands. Usually this third axis is the Z axis, but the user decides which axes have interpolated motion using the Select Interpolation Axes command. However, when 3-axes interpolation is turned on, motion commands will cause all three axes to move simultaneously in interpolated motion.

When 3-axes interpolation is turned on, two sets of parameters are still given for the third axis, but only the first set of parameters is followed. However, it is important that the second motion parameter for the third axis is set to 0 and the second speed parameter is set to 30.

An interpolation mode (2-axes or 3-axes) remains in effect until a subsequent interpolation command is sent.

☐ Examples

```
@07
@0e0
@0A  2700,3000,1000,2000,350,1000,0,1000
@0z1
@0A1000,100,2000,100,250,100,0,30
@0z0
```

This series of commands shows how you can flip between 2-axes and 3-axes interpolation.

The first line defines all three axes as being used. Line two then indicates that for 2-axes interpolation, the X and Y axes will have interpolated motion. Since 2-axes interpolation is in effect by default, line three then moves axes X and Y to absolute positions 2700 and 1000 in interpolated motion. After this, the Z axis is moved to position 350 and then position 0.

Line four selects the 3-axes interpolation mode since a 1 is given after the **z**.

Thus, in line five, all three axes will be moved (to absolute position 1000,2000,250) in simultaneous, interpolated motion. Notice that the last motion parameter was given a value of 0 and the last speed parameter a value of 30. Although these do not affect the motion, they must be included for the motion to occur.

Line six turns off the 3-axes interpolation. Hence all following motion commands will have 2-axes interpolation as described by line three above.

☐ See Also

Absolute Motion
Incremental Motion
Select Interpolation Axes

Command Summary/Index

Immediate Mode

Syntax	Page	Description
@0<axes>	**69**	Define Axes: x=1, y=2, z=4
@0R<axes>	**41**	Home: x=1, y=2, z=4, confirm completion
@0r<axes>	**41**	Home: x=1, y=2, z=4, confirm immediately
@0S	**71**	Start, confirm completion
@0s	**71**	Start, confirm immediately
@0d<Gx>, <Gy>, <Gz>	**63**	Set home speed
@0t	**75**	Start trace mode
@0A Sx, Gx, Sy.... Gz2	**46**	Incremental motion, confirm completion
@0a Sx, Gx, Sy... Gz2	**46**	Incremental motion, confirm immediately
@0M Sx, Gx, Sy... Gz2	**32**	Absolute motion
@0m Sx, Gx, Sy... Gz2	**32**	Absolute motion
@0k	**37**	Erase battery RAM
@0P	**43**	Identify axes' positions
@0B<Addr>,<Data>	**55**	Poke RAM address with data
@0c<Addr>	**54**	Read EPROM address
@0b<Addr>	**54**	Read RAM address
@0n<Axes>	**79**	Reset axes position to 0
@0eS	**58**	Select Interpolation axes/plane
{chr(254)}	**68**	Software reset
{chr(255)}	**73**	Stop Execution
@0i	**40**	Enter program mode
@0f	**61**	Circular interpolation direction
@0y...	**34**	Perform circular interpolation
@0z	**81**	3-axes interpolation: 1 = on, 0 = off

Program Mode

Syntax	Page	Description
0 Sx, Gx, Sy... Gz2	**46**	Incremental motion
1<Signal>	**60**	Send synchronization signal
2<Signal>,<Offset>	**77**	Wait for synchronization signal, branch if applicable
3<Number>,<Offset>	**49**	Make a loop, or branch if <number> = 0
4<Option>	**44**	Pulse, <Option> = number between 1 and 6
		1 = output ON (or IN)
		2 = output OFF (or OUT)
		3 = pulse for 0.5 sec.
		4 = wait for pulse
		5 = send pulse and wait for confirmation
		6 = wait for pulse and confirm 5<Time>
5 (Time)	**74**	Time delay, stated in 1/10 sec**.**
6 Sx, Gx, Sy... Gz2	**52**	Move until impulse
7<Axes>	**41**	Home motors, x=1, y=2, z=4
p<address> NV	**65**	Set outputs
n<Axes>	**79**	Set actual position as effective datum
mSx,Gx,Sy..... Gz2	**32**	Move absolute
eS	**58**	Select interpolation axes/plane: 0 = x/y, 1= x/z, 2 = y/z
f	**61**	Set direction of circular interpolation
y	**34**	Perform circular interpolation
z	**81**	3-axes interpolation, 1 = on, 0 = off
9	**39**	End program mode
o65531	**56**	Read Input Port

4

TROUBLESHOOTING

Guidelines For Troubleshooting

THESE GUIDELINES ARE MEANT FOR IDENTIFYING AND ISOLATING PROBLEMS ONLY. UNDER NO CIRCUMSTANCES SHOULD EQUIPMENT BE ADJUSTED OR TAKEN APART IN ANY WAY WITHOUT FIRST CONTACTING A TECHNO REPRESENTATIVE. IF YOU DO NOT FEEL THAT YOUR BACKGROUND MAKES YOU COMPETENT TO PERFORM THE CHECKS DISCUSSED HERE, DO NOT ATTEMPT THEM. INSTEAD, CALL TECHNO IMMEDIATELY.

No Holding Torque On The Motor

As soon as power is turned on to the controller, there should be holding torque on the motor. If this is not the case, there are a few checks you can make.

- Check to see if the Emergency Stop Button (see diagram on page 10) is pulled out. If not, turn clockwise to release.

- Check the motor cable:
  (1) Turn off power to the controller.

REMOVING A CABLE WITH POWER ON TO THE CONTROLLER CAN RESULT IN DAMAGE TO THE ELECTRONICS.

  (2) Switch the motor cable with one that is known to be functioning properly. If the holding torque is restored, the original cable was faulty.

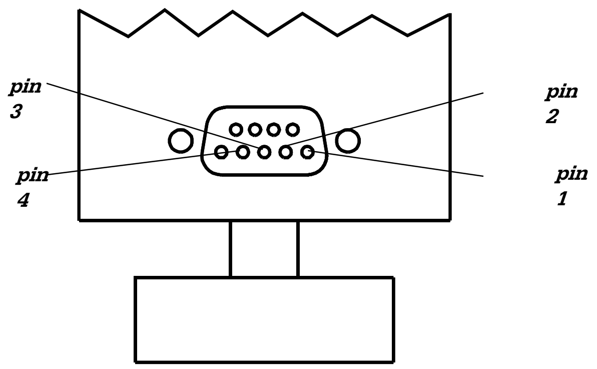
- Check that the stepper motor amplifier is plugged in (see page 10).

- Check that the stepper motor amplifier is not defective by switching it with one that is known to be functioning properly (see page 10).

- Check if a fuse is burned out:
  (1) Turn off power.
  (2) Remove the amplifier corresponding to the motor lacking holding torque (see page 10).
  (3) Examine fuses. If a filament within a fuse is broken, that fuse has burned out. Replace fuse. **Be sure to replace fuse with one of exactly the same type**. If fuse burns out again, call Techno.

- Check the motor circuits:

Use this close up of the motor housing as a guide to checking the motor.



(1) Using an OHM meter, test the resistance across each of the Motor Phases:

- Check the resistance between pins 1 and 2 on the connector (Motor Phase 1).
- Check the resistance between pins 3 and 4 on the connector (Motor Phase 2). Each phase should have a resistance between 0.6-1.2 ohms.
- A resistance less than this value indicates a short. Call Techno.
- A resistance much higher than this value indicates an open circuit or a broken wire. Call Techno.

(2) Using an OHM meter, check the resistance between pins 1 and 3.

- The resistance should be very high (at least 1M ohms) indicating an open circuit. If not, call Techno.

Holding Torque Present, But Motor Stalls During Travel

When a stepper motor skips steps, it generally stalls out and fails to complete the desired motion. It is very unusual for a stepper motor to only lose a few steps during a long motion. If the motor stalls, there are a number of possible causes:

- The slide may stall if you are trying to reach a speed beyond the limit of your machine. This limit depends upon the size of the amplifier, the size of the motor, your acceleration rate, the screw pitch of the machine, and the size of the load.

  Try running the slide at a slower speed. If the slide no longer stalls, then your original function was beyond the speed limit.

- Stepper motors exhibit resonances at certain velocities and loads. This generally happens around 200 to 500 steps/sec. This resonance can also occur when two or more motors are being moved similar amounts. For example, suppose you programmed the following:

  @ 0, 8000, 5000, 7700, 500

  The motor which is moving the shorter distance is here subject to a beat frequency at a rate about equal to the difference in the number of steps it takes relative to the longest axis. This beat frequency can also excite resonance.

  If you appear to have this problem, alter the velocity or the way the motors are being moved.

Inputs Or Outputs Are Not Functioning

The outputs are "open collector" types, and the inputs look for a ground switch. Make sure that you have connected the outputs and inputs correctly, and that the voltage for the malfunctioning group is correct. Refer to pages 12 to 16.

If an input or an output is not functioning, the first thing you should do is check to see if the device is hooked up correctly. If so then:

Connect the defective input or output line to another input/output to **determine whether the driver chip or the electronics connected to the signal are defective**.

If the input/output works at the new connection, there may be a problem with the electronics of the original input/output. Call Techno.

If the input/output does not work at the new connection, there may be a problem with the wire you are using to hook up the output/input, or with the device on the other end. Try using a new wire to hook up the inputs/outputs. If this solves the problem, your original cable was defective. Also try hooking up a new device to the I/O. If this solves the problem, your original device was defective.

Check your program to be sure that you have correctly turned on the input/output.

Check the voltage needed to activate the device; if the voltage is greater than that emitted by the output port, then the device will not work.

If you still have problem, the driver card may be defective, call Techno.

No Communication With The Controller

Check that the C108, C116, or C142 controller is connected to a working COM port:

- Refer to your PC Owner's Manual to confirm the location of your PC's COM Ports. Also check that manual to see if the COM Port you are using is functional and that you do not have to adjust the jumpers on your PC.

Check that the RS232 cable is firmly attached at both ends.

Check that the RS232 cable is functioning.

(1) Disconnect the RS232 cable from the controller, leaving it connected to the PC.

(2) Using a voltmeter, check the voltage across pins 1 and 2 of the cable. This voltage should be between -9 and -12 volts. **If it is not,** call Techno for further instructions.

5

TECHNICAL REFERENCE

Specifications Of Controller Card

Dimensions:	Standard Euro board 100mm x 160mm, front panel NOTE (2")
Power Supply:	5V, 3OOmA
Connector:	64 pin DIN connector
Inputs:	Computer reset reference switches 3 state driver
Outputs:	Current Inhibit Stop stepper motors Full/Half step selection 1 Output Port (3 state driver)
Serial Input:	Male DB9 connector

OPTIONAL

Part No.	**Description**
3356	Battery back up 3357
MAX232	serial chip 2794
RS232	cable from Interface card to 25 pin D-sub. connector for PC

Error Messages

0	= no error
2	= emergency stop or limit switch encountered
3	= illegal number of axes
4	= axis not defined
5	= syntax error
6	= out of memory
7	= illegal parameters
8	= illegal branch
c	= loop error
D	= illegal speed specified
G	= no data for execution in memory
z	= internal "SW" error
(CR)	= unexpected end of command

Self-Test Program

The following program can be used to test controller communications and the mechanics. The program will cause the controller card to move the X and Y axes and send the ASCII character set over the serial line until it receives a character which you type on the PC. A minimum of 50 characters are sent. Once a character is received from the PC, all subsequently received characters are echoed back to the host. This listing is provided only for reference. It is not necessary to input this program since it is preprogrammed on your controller card.

```
10 PRINT  "To perform the following test:"
20 PRINT "Turn on the controller while pressing the start button for"
30 PRINT" one second."
40 PRINT "Once the controller starts to send data to the screen, the data"
50 PRINT" can be stopped by hitting any key."
60 PRINT "After the controller stops sending characters, it will continue to"
70 PRINT "echo the characters typed on the keyboard. The controller should"
80 PRINT "be turned off and then stack on to reset it for normal operation."
90 PRINT "You can exit this BASIC program by typing -X-."
95 ON ERROR GOTO 400
100 OPEN "COMI:9600,N,8,1,RS,CSO,DSO CDO" AS #1 LEN-5500
110 IF LOC(l)>)THEN PRINT INPUT$(LOC(l),#);
120 A$-INKEY$ : IF A$ >',',THEN PRINT#l,A$; 125 IF A$-"X" THEN GOTO 200
130 GOTO 110
200 CLOSE
220 END
400 RESUME
```

Download Program

The following program is written in BASIC and is a general program to transfer a motion program to the controller card. This program first prompts the user for the name of the file containing the motion program and transmits the file, line by line, to the controller card. The software acknowledge is checked after each line is transmitted. If an error occurs, the error message is displayed and the downloading is aborted.

We have on occasion observed DEVICE I/O errors. If such an error does occur, it is best to just start the program again from the beginning.

```
10 OPEN"coml:9600,n,8,1,rs, cs, do, cd" AS #1 LEN-255
20 PRINT "Enter the name of the file to be transmitted-it
30 INPUT NA$
40 OPEN NA$ FOR INPUT AS#2
50 LINE INPUT#2,L$
60 PRINT #1,L$ : GOSUB 1000
70 PRINT L$
80 IF NOT (EOF(2) THEN GOTO 50
90 CLOSE
100 PRINT "File has successfully transmitted"
110 END
1000 REM this subroutine checks for the echo character
1010 REM if the echo is "O" then the transmission was successful
1020 REM if the echo is not "011 then an error has taken place
1030 IF LOC(1)>1 THEN GOTO 1000
1040 A$-INPUT$(1,1) : IF A$-"Olll THEN RETURN
1050 PRINT "Transmission Error - 1',A$
1060 STOP
```

This program can be used to test any of the commands presented in this manual. **The programs must be prepared on an ASCII word processor that does not add any special formatting characters**.

ASCII Codes

The following chart gives the ASCII character codes which are used as synchronization characters. Note that the decimal codes are those used within programs, whereas the characters appear on the Communications Screen when those codes are sent.

Decimal	**Character**
33	!
34	"
35	#
36	$
37	%
38	&
39	’
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:
59	;
60	<
61	=
62	>
63	?
64	Cannot be used
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z
91	[
92	\
93	]
94	^
95	_
96	`
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y
122	z
123	{
124	|

A Brief Lesson In Hexadecimals

Base Ten

Our normal counting system is in base ten. This means that the system uses ten unique symbols, namely 0,1,2,3,4,5,6,7,8, and 9, to represent numeric quantities. In base ten, the first digit to the right is the "units" digit. Once there are nine units, a second column is begun. This digit to the left is considered the "tens" digit, that is, it indicates how many tens are in the numeric quantity. This is also known as the "most significant digit" while the number to the right becomes the "least significant digit" telling the units left over after the tens are counted. Hence the number 18 is equal to 1 x 10 + 8 x 1 -- it has 1 ten in it and 8 ones left over. Once there are nine values in both columns (99), the third column or "hundreds" column is created, and that digit tells how many hundreds are in the value etc.

Base Sixteen

Hexadecimals numbers are in base sixteen. Hence the system uses sixteen unique symbols to represent numeric quantities. These symbols correspond to our decimal symbols as follows:

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Since some symbols are shared between decimal and hexadecimal numbers, hexadecimal numbers (known as hex values) are usually followed by the suffix "H" Hence 9 would be a decimal value while 9H would be the equivalent hex value.

In hex values, the digit farthest to the right is again the "units" digit. However, since this is base sixteen, and there are sixteen symbols, the next column is not started until fifteen units fill the first column. Hence the second digit to the left will be known as the "sixteens" digit, indicating how many sixteens are in the numeric quantity. This again is considered the most significant digit, with the digit to the right becoming the least significant digit and giving the number of units left over after the sixteens are counted. Hence in hexadecimal notation the number 18H is equal to 1 x 16 + 8 x 1 -- it has one sixteen in it and eight ones left over (and thus would be equal to the decimal value 24).

The largest two digit number in hexadecimal notation would thus be FFH which means that there are fifteen units in each column. The decimal value of this number would be 15 x 16 + 15 x 1 or 255. Hence, when one more unit is added, a third column, the "two-hundred fifty sixes" column will be created, telling how many two hundred fifty sixes are in the value etc.

Two digit Hex values are important because computers work with the fundamental numeric unit called the byte. Bytes can have decimal values anywhere from 0 to 255 which happens to coincide exactly with the range of two-digit Hex numbers (0H to FFH).

Converting Between Bases

To convert any base ten (decimal) number between 0 and 255 to a Hex value, divide the number by 16. The integer part of the quotient becomes the most significant digit (in the left column) while the remainder becomes the least significant digit (the number in the right column). For example:

To convert 143 into Hex, divide 143 by 16:

```
     ___8_   Remainder 15 (F in Hex symbols)
  16)143
```

Hence the Hex value would be 8FH.

To convert a decimal number larger than 255 to a Hex value, use the following procedure:

(1) Divide the number by 16 and record the remainder. This remainder is the digit in the furthest right column of the hex number (the least significant digit).

(2) If the dividend of the result is greater than zero, then repeat step one. The next remainder goes in the next column to the left.

(3) Continue until the dividend is zero. The last remainder will be the digit in the column furthest to the left (the most significant digit).

For example, to convert 8975 to Hex:

```
   __560       Remainder 15 (F in Hex digits, the least significant digit)
16)8975

   __35        Remainder 0
16)560

   __2         Remainder  3
16)35

   __0         Remainder 2 (the most significant digit)
16)2
```

Hence the Hex value would be 230FH

To convert Hex numbers into decimal numbers, remember that each column in a Hex number represents the 16^{n-1} units where n is the number of the column counting from the right. The value for a column is then the digit in that column multiplied by 16^{n-1} , and the decimal equivalent of the Hex number is the sum of these values.

For example to convert FA2DH to decimal numbers:

F	=	$15 \times 16^{4-1}$	=	$15x16^3$	=	15x4096	=	61440
A	=	$10 \times 16^{3-1}$	=	$15x16^2$	=	15x256	=	3840
2	=	$2 \times 16^{2-1}$	=	$15x16^1$	=	15x16	=	240
D	=	$13 \times 16^{1-1}$	=	$15x16^0$	=	13x1	=	13
						FA2DH	=	65533

2's Complement

The hexadecimal form of a 2's complement positive decimal number in the range 0 to 8,388,607 is calculated as described above.

The hexadecimal form of a 2's complement negative decimal number X can be calculated by finding the Hex value of 16777216-X for any number in the range -1 to -8,388,608.

The decimal equivalent of a 2's complement positive hexadecimal number (in the range 0 to EFFFFF) can likewise be derived in the normal manner as described above.

The decimal equivalent of a 2's complement negative hexadecimal number (in the range F00000 to FFFFFF) can be found by again normally converting the number to decimal form and then subtracting 16777216 from the result.

6

APPENDIX

Connector Pinout For Interface Card 4.*, 5.*

		A	C		
NC		1	1		NC
NC		2	2		NC
NC		3	3		NC
NC		4	4		NC
NC		5	5		NC
NC		6	6		NC
F/H X-Axis	(A)	7	7		NC
F/H Z-Axis	(A)	8	8	(A)	F/H Y-Axis
Home Y-Axis	(E)	9	9	(E)	HOME X-Axis
NC		10	10	(E)	HOME Z-Axis
Pulse Inhibit X	(A)	11	11	(A)	Pulse Inhibit Y
Pulse Inhibit Z	(A)	12	12		NC
NC		13	13		NC
NC		14	14		NC
NC		15	15		NC
NC		16	16	(A)	Direction X
Pulse X	(A)	17	17	(A)	Direction Z
Pulse Z	(A)	18	18	(A)	Direction Y
Pulse Y	(A)	19	19		NC
Current Reduction Z	(A)	20	20	(A)	Current Reduction Y
Current Reduction X	(A)	21	21		NC
NC		22	22		NC
NC		23	23		NC
NC		24	24		NC
Overtravel Z	(E)	25	25	(E)	Overtravel Y
Overtravel X	(E)	26	26		P1.0
P1.0		27	27		P1.0
NC		28	28		μP-Reset
NC		29	29		NC
+5V		30	30		+5V
NC		31	31		internal used
Gnd		32	32		Gnd

NC = no connection
(A) = signal output
(E) = signal input