

---

Hardware

# Model 5000 Hardware Guide

**KEITHLEY**

# WARRANTY

## Hardware

Keithley Instruments, Inc. warrants that, for a period of one (1) year from the date of shipment (3 years for Models 2000, 2001, 2002, and 2010), the Keithley Hardware product will be free from defects in materials or workmanship. This warranty will be honored provided the defect has not been caused by use of the Keithley Hardware not in accordance with the instructions for the product. This warranty shall be null and void upon: (1) any modification of Keithley Hardware that is made by other than Keithley and not approved in writing by Keithley or (2) operation of the Keithley Hardware outside of the environmental specifications therefore.

Upon receiving notification of a defect in the Keithley Hardware during the warranty period, Keithley will, at its option, either repair or replace such Keithley Hardware. During the first ninety days of the warranty period, Keithley will, at its option, supply the necessary on site labor to return the product to the condition prior to the notification of a defect. Failure to notify Keithley of a defect during the warranty shall relieve Keithley of its obligations and liabilities under this warranty.

## Other Hardware

The portion of the product that is not manufactured by Keithley (Other Hardware) shall not be covered by this warranty, and Keithley shall have no duty of obligation to enforce any manufacturers' warranties on behalf of the customer. On those other manufacturers' products that Keithley purchases for resale, Keithley shall have no duty of obligation to enforce any manufacturers' warranties on behalf of the customer.

## Software

Keithley warrants that for a period of one (1) year from date of shipment, the Keithley produced portion of the software or firmware (Keithley Software) will conform in all material respects with the published specifications provided such Keithley Software is used on the product for which it is intended and otherwise in accordance with the instructions therefore. Keithley does not warrant that operation of the Keithley Software will be uninterrupted or error-free and/or that the Keithley Software will be adequate for the customer's intended application and/or use. This warranty shall be null and void upon any modification of the Keithley Software that is made by other than Keithley and not approved in writing by Keithley.

If Keithley receives notification of a Keithley Software nonconformity that is covered by this warranty during the warranty period, Keithley will review the conditions described in such notice. Such notice must state the published specification(s) to which the Keithley Software fails to conform and the manner in which the Keithley Software fails to conform to such published specification(s) with sufficient specificity to permit Keithley to correct such nonconformity. If Keithley determines that the Keithley Software does not conform with the published specifications, Keithley will, at its option, provide either the programming services necessary to correct such nonconformity or develop a program change to bypass such nonconformity in the Keithley Software. Failure to notify Keithley of a nonconformity during the warranty shall relieve Keithley of its obligations and liabilities under this warranty.

## Other Software

OEM software that is not produced by Keithley (Other Software) shall not be covered by this warranty, and Keithley shall have no duty or obligation to enforce any OEM's warranties on behalf of the customer.

## Other Items

Keithley warrants the following items for 90 days from the date of shipment: probes, cables, rechargeable batteries, diskettes, and documentation.

## Items not Covered under Warranty

This warranty does not apply to fuses, non-rechargeable batteries, damage from battery leakage, or problems arising from normal wear or failure to follow instructions.

## Limitation of Warranty

This warranty does not apply to defects resulting from product modification made by Purchaser without Keithley's express written consent, or by misuse of any product or part.

## Disclaimer of Warranties

EXCEPT FOR THE EXPRESS WARRANTIES ABOVE KEITHLEY DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEITHLEY DISCLAIMS ALL WARRANTIES WITH RESPECT TO THE OTHER HARDWARE AND OTHER SOFTWARE.

## Limitation of Liability

KEITHLEY INSTRUMENTS SHALL IN NO EVENT, REGARDLESS OF CAUSE, ASSUME RESPONSIBILITY FOR OR BE LIABLE FOR: (1) ECONOMIC, INCIDENTAL, CONSEQUENTIAL, INDIRECT, SPECIAL, PUNITIVE OR EXEMPLARY DAMAGES, WHETHER CLAIMED UNDER CONTRACT, TORT OR ANY OTHER LEGAL THEORY, (2) LOSS OF OR DAMAGE TO THE CUSTOMER'S DATA OR PROGRAMMING, OR (3) PENALTIES OR PENALTY CLAUSES OF ANY DESCRIPTION OR INDEMNIFICATION OF THE CUSTOMER OR OTHERS FOR COSTS, DAMAGES, OR EXPENSES RELATED TO THE GOODS OR SERVICES PROVIDED UNDER THIS WARRANTY.



**Keithley Instruments, Inc.** • 28775 Aurora Road • Cleveland, OH 44139 • 440-248-0400 • Fax: 440-248-6168 • <http://www.keithley.com>

<b>CHINA:</b>	<b>Keithley Instruments China</b> • Yuan Chen Xin Building, Room 705 • 12 Yumin Road, Dewai, Madian • Beijing 100029 • 8610-62022886 • Fax: 8610-62022892
<b>FRANCE:</b>	<b>Keithley Instruments SARL</b> • BP 60 • 3 Allée des Garays • 91122 Palaiseau Cédex • 33-1-60-11-51-55 • Fax: 33-1-60-11-77-26
<b>GERMANY:</b>	<b>Keithley Instruments GmbH</b> • Landsberger Strasse 65 • D-82110 Germering, Munich • 49-89-8493070 • Fax: 49-89-84930759
<b>GREAT BRITAIN:</b>	<b>Keithley Instruments, Ltd.</b> • The Minster • 58 Portman Road • Reading, Berkshire RG30 1EA • 44-1189-596469 • Fax: 44-1189-575666
<b>ITALY:</b>	<b>Keithley Instruments SRL</b> • Viale S. Gimignano 38 • 20146 Milano • 39-2-48303008 • Fax: 39-2-48302274
<b>NETHERLANDS:</b>	<b>Keithley Instruments BV</b> • Avelingen West 49 • 4202 MS Gorinchem • 31-(0)183-635333 • Fax: 31-(0)183-630821
<b>SWITZERLAND:</b>	<b>Keithley Instruments SA</b> • Kriesbachstrasse 4 • 8600 Dübendorf • 41-1-8219444 • Fax: 41-1-8203081
<b>TAIWAN:</b>	<b>Keithley Instruments Taiwan</b> • 1FL., 85 Po Ai Street • Hsinchu, Taiwan • 886-3-572-9077 • Fax: 886-3-572-9031

# Model 5000 Hardware Guide

©1998, Keithley Instruments, Inc.  
All rights reserved.  
Cleveland, Ohio, U.S.A.  
Second Printing, August 1998  
Document Number: 81830 Rev. B

# Manual Print History

The print history shown below lists the printing dates of all Revisions and Addenda created for this manual. The Revision Level letter increases alphabetically as the manual undergoes subsequent updates. Addenda, which are released between Revisions, contain important change information that the user should incorporate immediately into the manual. Addenda are numbered sequentially. When a new Revision is created, all Addenda associated with the previous Revision of the manual are incorporated into the new Revision of the manual. Each new Revision includes a revised copy of this print history page.

Revision A (Document Number 81830) .....	July 1996
Revision B (Document Number 81830) .....	August 1998

# Safety Precautions

---

The following safety precautions should be observed before using this product and any associated instrumentation. Although some instruments and accessories would normally be used with non-hazardous voltages, there are situations where hazardous conditions may be present.

This product is intended for use by qualified personnel who recognize shock hazards and are familiar with the safety precautions required to avoid possible injury. Read the operating information carefully before using the product.

The types of product users are:

**Responsible body** is the individual or group responsible for the use and maintenance of equipment, and for ensuring that operators are adequately trained.

**Operators** use the product for its intended function. They must be trained in electrical safety procedures and proper use of the instrument. They must be protected from electric shock and contact with hazardous live circuits.

**Maintenance personnel** perform routine procedures on the product to keep it operating, for example, setting the line voltage or replacing consumable materials. Maintenance procedures are described in the manual. The procedures explicitly state if the operator may perform them. Otherwise, they should be performed only by service personnel.

**Service personnel** are trained to work on live circuits, and perform safe installations and repairs of products. Only properly trained service personnel may perform installation and service procedures.

Exercise extreme caution when a shock hazard is present. Lethal voltage may be present on cable connector jacks or test fixtures. The American National Standards Institute (ANSI) states that a shock hazard exists when voltage levels greater than 30V RMS, 42.4V peak, or 60VDC are present. **A good safety practice is to expect that hazardous voltage is present in any unknown circuit before measuring.**

Users of this product must be protected from electric shock at all times. The responsible body must ensure that users are prevented access and/or insulated from every connection point. In some cases, connections must be exposed to potential human contact. Product users in these circumstances must be trained to protect themselves from the risk of electric shock. If the circuit is capable of operating at or above 1000 volts, **no conductive part of the circuit may be exposed.**

As described in the International Electrotechnical Commission (IEC) Standard IEC 664, digital multimeter measuring circuits (e.g., Keithley Models 175A, 199, 2000, 2001, 2002, and 2010) are Installation Category II. All other instruments' signal terminals are Installation Category I and must not be connected to mains.

Do not connect switching cards directly to unlimited power circuits. They are intended to be used with impedance limited sources. NEVER connect switching cards directly to AC mains. When connecting sources to switching cards, install protective devices to limit fault current and voltage to the card.

Before operating an instrument, make sure the line cord is connected to a properly grounded power receptacle. Inspect the connecting cables, test leads, and jumpers for possible wear, cracks, or breaks before each use.

For maximum safety, do not touch the product, test cables, or any other instruments while power is applied to the circuit under test. ALWAYS remove power from the entire test system and discharge any capacitors before: connecting or disconnecting cables or jumpers, installing or removing switching cards, or making internal changes, such as installing or removing jumpers.


Do not touch any object that could provide a current path to the common side of the circuit under test or power line (earth) ground. Always make measurements with dry hands while standing on a dry, insulated surface capable of withstanding the voltage being measured.


Do not exceed the maximum signal levels of the instruments and accessories, as defined in the specifications and operating information, and as shown on the instrument or test fixture panels, or switching card.


When fuses are used in a product, replace with same type and rating for continued protection against fire hazard.

Chassis connections must only be used as shield connections for measuring circuits, NOT as safety earth ground connections.

If you are using a test fixture, keep the lid closed while power is applied to the device under test. Safe operation requires the use of a lid interlock.

If a  screw is present, connect it to safety earth ground using the wire recommended in the user documentation.

The  symbol on an instrument indicates that the user should refer to the operating instructions located in the manual.

The  symbol on an instrument shows that it can source or measure 1000 volts or more, including the combined effect of normal and common mode voltages. Use standard safety precautions to avoid personal contact with these voltages.

The **WARNING** heading in a manual explains dangers that might result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading in a manual explains hazards that could damage the instrument. Such damage may invalidate the warranty.

Instrumentation and accessories shall not be connected to humans.

Before performing any maintenance, disconnect the line cord and all test cables.

To maintain protection from electric shock and fire, replacement components in mains circuits, including the power transformer, test leads, and input jacks, must be purchased from Keithley Instruments. Standard fuses, with applicable national safety approvals, may be used if the rating and type are the same. Other components that are not safety related may be purchased from other suppliers as long as they are equivalent to the original component. (Note that selected parts should be purchased only through Keithley Instruments to maintain accuracy and functionality of the product.) If you are unsure about the applicability of a replacement component, call technical support for information.

To clean the instrument, use a damp cloth or mild, water based cleaner. Clean the exterior of the instrument only. Do not apply cleaner directly to the instrument or allow liquids to enter or spill on the instrument.

---

# About this manual

## Quality control

Keithley Instruments manufactures quality and versatile products, and we want our documentation to reflect that same quality. We take great pains to publish manuals that are informative and well organized. We also strive to make our documentation easy to understand for the novice as well as the expert.

If you have comments or suggestions about how to make this (or other) manuals easier to understand, or if you find an error or an omission, please fill out and mail the reader response card at the end of this manual (postage is prepaid).

## Conventions

### Procedural

Keithley Instruments uses various conventions throughout this manual. You should become familiar with these conventions as they are used to draw attention to items of importance and items that will generally assist you in understanding a particular area.

***WARNING***      **A warning is used to indicate that an action must be done with great care. Otherwise, personal injury may result.**

***CAUTION***      **A caution is used to indicate that an action may cause minor equipment damage or the loss of data if not performed carefully.**

***NOTE***            *A note is used to indicate important information needed to perform an action or information that is nice-to-know.*

When referring to pin numbering, pin 1 is always associated with a square solder pad on the actual component footprint.

### Notational

A forward slash (/) preceding a signal name denotes an active LOW signal. This is a standard Intel convention.

Caret brackets (<>) denote keystrokes. For instance <Enter> represents carriage-return-with-line-feed keystroke, and <Esc> represents an escape keystroke.

Driver routine declarations are shown for C and BASIC (where applicable).

Hungarian notation is used for software parameters. In other words, the parameter type is denoted by a one or two letter lower case prefix:

c	character, signed or unsigned
s	short integer, signed
w	short integer, unsigned
l	long integer, signed
dw	long integer, unsigned

For example, wBoardAddr would be an unsigned short integer parameter.

---

An additional `p` prefix before the type prefix indicates that the parameter is being passed by reference instead of by value. (A pointer to the variable is being passed instead of the variable itself).

For example, `pwErr` would be an unsigned short integer parameter passed by reference.

This notation is also used in BASIC although no distinction between signed and unsigned variables exists.

In BASIC, all parameters also have a type suffix:

\$	character, signed or unsigned
%	integer, signed or unsigned
&	long integer, signed or unsigned

Routine names are printed in bold font when they appear outside of function declarations, e.g., **ReadStatus**.

Parameter names are printed in italics when they appear outside of function declarations, e.g., *sControls*.

Constants are defined with all caps, e.g., `ALL_AXES`. Underscores {`_`} must be replaced by periods {`.`} for use with BASIC.

Combinational logic and hexadecimal notation is in C convention in many cases. For example, the hexadecimal number `7Ch` is shown as `0x7C`.

C relational operators for OR and AND functions — “`|`” and “`&&`” — are used to minimize the confusion associated with grammar.



# Table of Contents

---

## 1 Introduction and Installation

Description of the 5000 .....	1-2
Technical specifications .....	1-2
Installation .....	1-3
Power .....	1-3
W7, board base address.....	1-3
W9 to W11, clock speed select .....	1-5
W8, interrupt select .....	1-6
W1 to W6, opto power (bus) .....	1-7
Connector pinouts .....	1-8

## 2 Operation and Programming

Theory of operation .....	2-2
Loading the Axis A counter .....	2-2
Port locations .....	2-3
Addressing an onboard port .....	2-3
Programming .....	2-4
Reading from an onboard port .....	2-4
Writing the controller internal registers .....	2-4
Writing to an onboard port .....	2-4
Reading the stepper controller status port .....	2-5
Reading the state buffer .....	2-5
Sync latch .....	2-5
Reset latch .....	2-6
Command buffer .....	2-6
Operating mode selection .....	2-6
Control mode selection .....	2-8
Data register selection .....	2-9
Output mode selection .....	2-10
User-accessible registers .....	2-11
Register descriptions .....	2-11

## 3 Interrupt Control

Description of interrupt control .....	3-2
Interrupt Request Rgtr (IRR), In-Service Rgtr (ISR) .....	3-3
Priority Resolver (PR) .....	3-3
Interrupt Mask Register (IMR) .....	3-3
Interrupt Output (INT) .....	3-3
PIC operation .....	3-4
Interrupt sequence, 80x86/80x88 mode .....	3-4
End-of-Interrupt command .....	3-4
Completing an interrupt .....	3-5
Operating modes .....	3-5
Fully nested mode .....	3-5
Special mask mode .....	3-5
Specific rotation (specific priority) .....	3-5
Automatic rotation (equal priority) .....	3-5
Non-vectored mode (poll command) .....	3-5
PIC programming .....	3-6
Initialization Command Words (ICW) .....	3-6
ICW1 format and description .....	3-7
ICW4 format and description .....	3-7
Operation Command Words (OCW) .....	3-8
OCW1 format and description .....	3-9
OCW2 format and description .....	3-9
OCW2 commands .....	3-9
OCW3 format and description .....	3-10

## A Typical Operation Procedures

Initialization flow chart .....	A-2
Setting speed data .....	A-3
High speed preset mode .....	A-4
Constant speed preset mode .....	A-5
High speed continuous mode .....	A-6
Constant speed continuous mode .....	A-7
High speed return to home .....	A-8
Constant speed return to home .....	A-9
Speed change during operation .....	A-10
Additional operating parameters .....	A-11

## B Typical High Speed Preset Mode Calculations

Typical high speed preset mode calculations .....	B-2
Determine R2 to R7 from given values .....	B-3
Example 1 .....	B-4
Example 2 .....	B-5

## C PC I/O and Interrupt Mapping

PC I/O map .....	C-2
------------------	-----

<b>D</b>	<b>Software Read/Write Technique</b>	
	Software read/write technique .....	D-2
<b>E</b>	<b>Tech Bulletins and Application Notes</b>	
	Optimizing the scale factor, $n$ .....	E-2
<b>F</b>	<b>Revision History</b>	
	Revision/ .....	F-2
	Revision A .....	F-2
	Revision B .....	F-2
	Revision C .....	F-2
	Revision D .....	F-2
	Revision E .....	F-3
	Revision F .....	F-3
	Revision G .....	F-3
	Revision H .....	F-3
<b>G</b>	<b>Introduction to the Model 9011 Motion Simulator</b>	
	Features .....	G-2
	Application .....	G-2
	General description .....	G-2
	Technical specifications .....	G-2
	Operation .....	G-3
	Servo systems .....	G-3
	Stepper systems .....	G-3
	Encoders .....	G-3
	Pin assignments .....	G-4
	Common applications for the Model 9011 .....	G-5
<b>H</b>	<b>Circuit Diagrams</b>	

# List of Illustrations

---

## 1 Introduction and Installation

Figure 1-1	Model 5000 functional block diagram .....	1-2
Figure 1-2	5000 board layout .....	1-4
Figure 1-3	Clock speed select jumpers .....	1-5
Figure 1-4	Interrupt select jumper .....	1-6
Figure 1-5	Opto power select jumpers .....	1-7
Figure 1-6	Optoisolation on the 5000 .....	1-8
Figure 1-7	J1 pins allocated by axis .....	1-9

## 2 Operation and Programming

Figure 2-1	Acceleration profile .....	2-12
Figure 2-2	Deceleration profile .....	2-13

## 3 Interrupt Control

Figure 3-1	PIC block level diagram .....	3-2
Figure 3-2	PIC initialization sequence .....	3-6
Figure 3-3	PIC ICW format .....	3-7
Figure 3-4	PIC OCW format .....	3-8

## G Introduction to the Model 9011 Motion Simulator

Figure G-1	Analog or $\pm$ PWM application .....	G-5
Figure G-2	Pulse and direction .....	G-6
Figure G-3	Using the 9011 to isolate encoder problems .....	G-6
Figure G-4	Stepper system .....	G-7
Figure G-5	Driving a servo amplifier .....	G-7
Figure G-6	Driving a stepper motor system .....	G-8

# List of Tables

---

## 1 Introduction and Installation

Table 1-1	Voltage requirements for the 5000 .....	1-3
Table 1-2	Selecting the base address .....	1-3
Table 1-3	Selecting the clock speed with W9 to W11 .....	1-5
Table 1-4	Selecting the PC bus interrupt request .....	1-6
Table 1-5	Selecting opto power .....	1-7
Table 1-6	Connector J1 pin definitions .....	1-9

## 2 Operation and Programming

Table 2-1	Onboard port map .....	2-3
Table 2-2	Command mode selection .....	2-6
Table 2-3	Operating mode selection .....	2-7
Table 2-4	Data register selection .....	2-9
Table 2-5	Controller register reference .....	2-11

## 3 Interrupt Control

Table 3-1	Interrupt code .....	3-4
-----------	----------------------	-----

## C PC I/O and Interrupt Mapping

Table C-1	PC I/O map .....	C-2
Table C-2	PC interrupt map .....	C-3

## G Introduction to the Model 9011 Motion Simulator

Table G-1	Model 9011 connector definitions .....	G-4
Table G-2	9011 Connections for common applications .....	G-5



**1**

# Introduction and Installation

## Description of the 5000

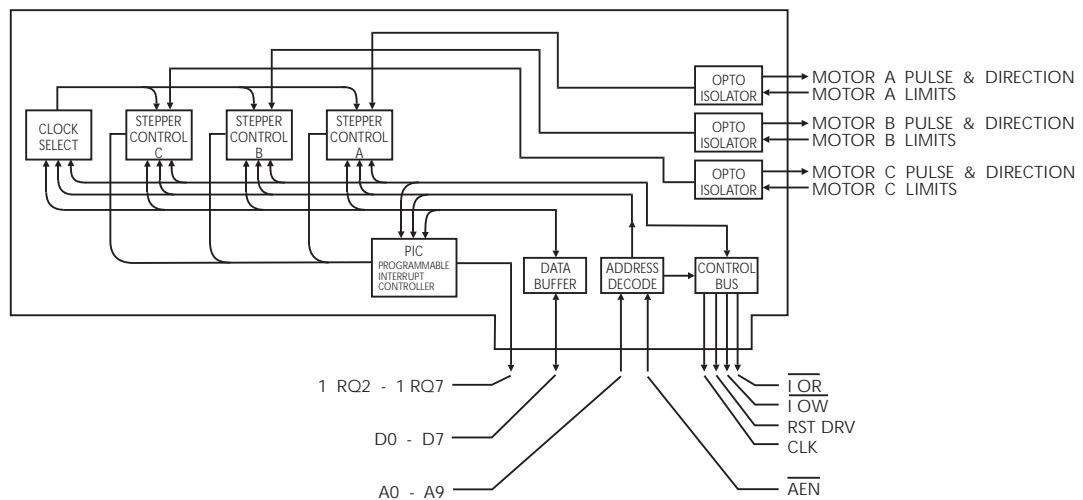
The 5000 Stepper Motor Controller card allows a PC/XT/AT compatible computer to control three independent stepper motor drivers. This allows you to perform complex motion control profiling routines.

Each axis of control provides five limit inputs — two stop limits, two deceleration limits, and one home limit. Outputs include pulse/direction and hold. The pulse/direction output allows a rate of up to 240,000 pulses per second. All inputs and outputs are optically isolated to minimize noise at the card cage and protect the card from induced voltage transients.

Operation of the 5000 is controlled through user-accessible internal registers. Data registers enable specification of the following values: number of steps, low speed rate, high speed rate, acceleration rate, deceleration rate, and rampdown point. Intelligent controller chips (one per axis) enable programmable velocity profiling, including direction. You may select from four programming modes and eight operating modes.

Figure 1-1 shows a functional block diagram of the Model 5000.

Figure 1-1  
Model 5000 functional block diagram



## Technical specifications

<b>Compatibility:</b>	PC/XT/AT compatible computers
<b>Card Dimensions:</b>	13.13 x 4.200 x 0.500 inches
<b>Operating Range:</b>	0 degrees to 70 degrees C
<b>Voltage:</b>	Refer to Table 1-1
<b>Mating Connectors:</b>	37 Pin D Sub (J1)      Amp 206802-1 Ansley 609-37D Berg 66167-237

Table 1-1  
Voltage requirements for the 5000

Voltage	Nominal Current	Maximum Current	Notes
PC Bus: +5V	1.3A	2.5A	Using bus +5V to power optos
PC Bus: +12V	0	0	
PC Bus: +5V	1.0A	2.0A	Using bus +12V to power optos
PC Bus: +12V	0.4A	0.8A	
Auxiliary: +5V	0.4A	0.8A	Using auxiliary +5V to power optos
Auxiliary: +12V	0	0	
Auxiliary: +5V	0	0	Using auxiliary +12V to power optos
Auxiliary: +12V	0.4A	0.8A	

## Installation

Figure 1-2 shows the 5000 layout with jumper and connector locations. Jumpers consist of unshrouded headers and shorting connectors. The jumper options are explained below.

To determine the correct jumper configuration for the system, follow the guidelines given below. The † symbol indicates default jumper locations.

### Power

The 5000 requires +5 volts from the PC Bus. Motor power must come from an external supply capable of supplying motor current plus approximately 300mA of opto current. Provisions are made to supply opto current from the PC Bus. However, using voltage from the PC Bus to drive the optoisolators eliminates isolation. Therefore, it is not recommended for permanent use.

### W7, board base address

Jumper W7 determines the upper 4-bit nibble of the board I/O address according to Table 1-2.

Hex switches SW1 and SW2 determine the lower 8 bits of the address, with SW2 representing the most significant nibble (MSN) and SW1 representing the least significant nibble (LSN). Since the card occupies two successive I/O ports, only the even settings of the LSN switch are used.

Table 1-2  
Selecting the base address

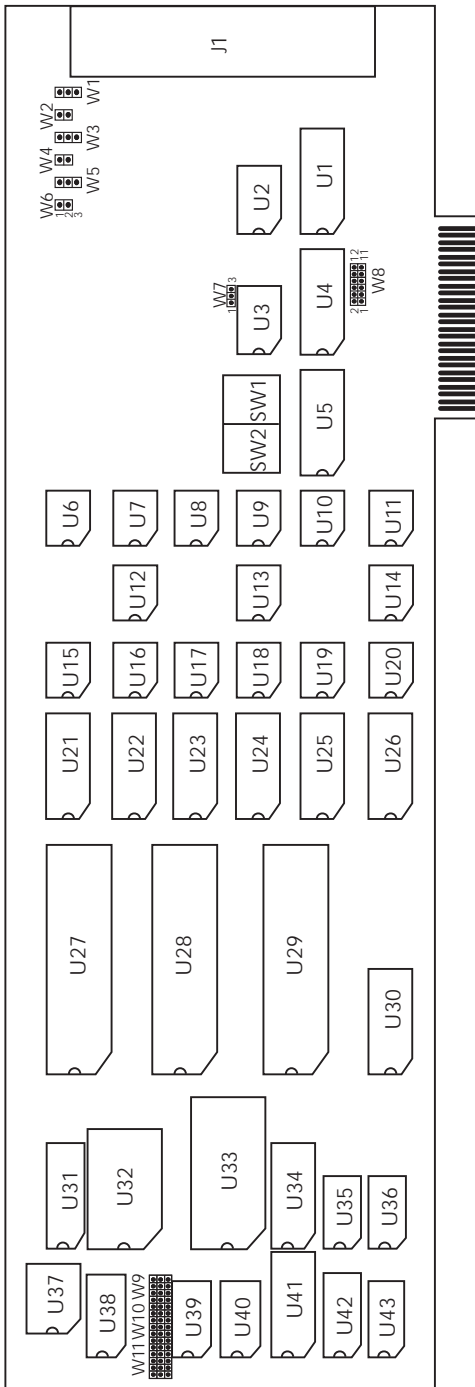
W7	I/O Address
(1-2)	3xxh†
(2-3)	2xxh

† Default setting.

Note: This jumper determines the upper 4-bit nibble of the board I/O address.



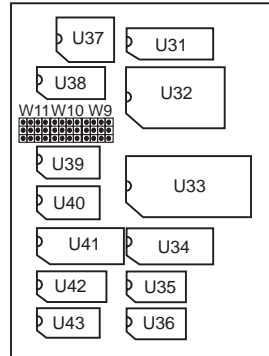
Figure 1-2  
5000 board layout



Note: Passive devices are not shown to improve clarity.

## W9 to W11, clock speed select

Figure 1-3  
Clock speed select jumpers



Note: Jumpers are viewed with the bus edge connector down and to the right.

Figure 1-3 shows the physical locations. W9 – W11 (W11 — Axis A, W10 — Axis B, and W11 — Axis C) are used to select the base clock rate ( $f_{clock}$ ) used by each stepper motor controller. The  $f_{clock}$  rate supplied to each controller determines the available step rate. It is also used in the formula in Appendix B to determine the actual pulse rate. Select a clock rate from Table 1-3.

Table 1-3  
Selecting the clock speed with W9 to W11

W9, W10, W11	Frequency (MHz)
(1-2)	0.625
(3-4)	1.25
(5-6)	2.50
(7-8)	5.00†

† Default setting.

Note: To obtain the maximum rate of 240,000 pulses per second, use the default clock setting. These jumpers select the base clock rate used by axes A to C.

The default setting, 5MHz, may be altered to allow slower step rates. Examples and formulas for setting defaults are shown in Section 2 and Appendix E of this manual.

## W8, interrupt select

Figure 1-4 shows the physical location. W8 selects the PC Bus interrupt request line used for interrupt operation according to Table 1-4.

Figure 1-4  
Interrupt select jumper

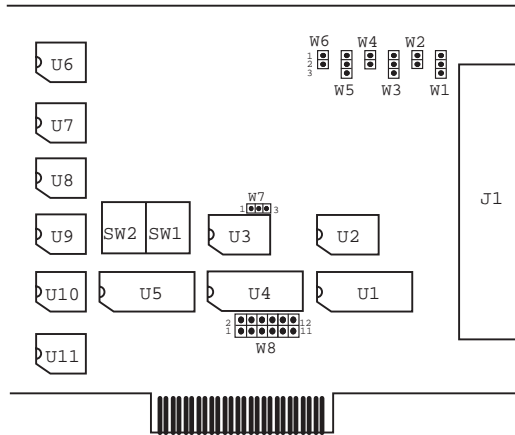


Table 1-4  
Selecting the PC bus interrupt request

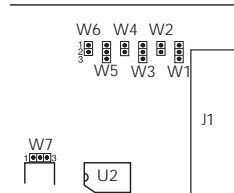
W8	Interrupt Request	Description
(1-2)	IRQ2	unused†
(3-4)	IRQ3	unused
(5-6)	IRQ4	Serial Port Card
(7-8)	IRQ5	unused
(9-10)	IRQ6	Diskette Adapter Card
(11-12)	IRQ7	Parallel Port Card

† Default setting. For applications not requiring interrupts, you may remove the strap from W8.

Note: The requests listed are defaults in most PC systems.

## W1 to W6, opto power (bus)

Figure 1-5  
Opto power select jumpers



Note: Jumpers are viewed with the bus edge connector down and to the right.

Table 1-5  
Selecting opto power

Axis	Jumper	Strap	Description
A	W1	(1-2)	+12V
		(2-3)	+5V
	W2	(1-2)	Ground
B	W3	(1-2)	+12V
		(2-3)	+5V
	W4	(1-2)	Ground
C	W5	(1-2)	+12V
		(2-3)	+5V
	W6	(1-2)	Ground

Note: You can select from either internal or external opto power.

Figure 1-5 shows the physical locations. W1 through W6 are provided for customer convenience during development (W1, W2 — Axis A; W3, W4 — Axis B; W5, W6 — Axis C). They enable the use of bus power for the optoisolators according to Table 1-5.

These jumpers allow the bus to supply voltage required by the output side of the optoisolators on all lines interfacing with the connector at the rear of the card. However, this is not recommended because damaging spikes can be induced on the output interface and will conduct through the computer power supply. These jumpers should be removed. Use an external voltage source to power the output side of the optos. To use voltage supplied by the motor, connect the positive connection to the power pins on the I/O connector and the negative connection to the ground pins on the I/O connector.

Current limiting resistors protect the output side of the optoisolators. Resistors must be calculated as follows:

$$R_n = \frac{V - 1.5}{I}$$

where: I = 10mA

The 5000 is set up at the factory to run at 5V. To run at 12V (assuming 12V is really 13V, worst case), R<sub>n</sub> will be:

$$\frac{13 - 1.5}{0.010} = 1150\Omega$$

The next standard 10% value is 2.2K. R3–R12 are all ready at 2.2K, so they can be held at that value.

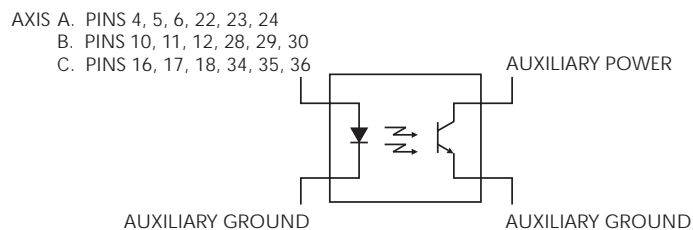
Change RP3, RP5, and RP7 from 680 ohms to at least 1.2K. Use at least 1/4W resistors to reduce Mean Time Between Failures (MTBF).

## Connector pinouts

The following subsection contains connector pinout information. Suggested mating connector part numbers are listed in paragraph *technical specifications*. Table 1-6 shows the pin definitions for connector J1.

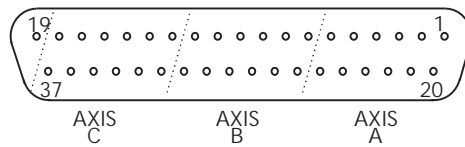
These are active low inputs for use with normally open switches. To use normally closed switches or optointerrupter modules, ICs — U22, U24, and U26 — must be changed from 74LS244 to 74LS240. These ICs are socketed to make this change easy. Figure 1-6 shows the optoisolation method used on the 5000. Figure 1-7, partitioned by axis, and Tables 1-6 and 1-7 show the pinout of connector J1.

Figure 1-6  
**Optoisolation on the 5000**



Note: Limits, Home, Rampdown, and INS are optoisolated.

Figure 1-7  
J1 pins allocated by axis



Note: The view is looking into the connector.

Table 1-6  
Connector J1 pin definitions

Pins			Name	Description
Axis A	Axis B	Axis C		
1	7	13	Auxiliary +12V	12V input optoisolator supply unless W1, W3, and W5 are jumpered to use PC Bus power (this is not recommended)
2	8	14	Direction	Direction output
/3	/9	/15	User definable	OTS signal output (see output Data Register Selection)
/4	/10	/16	+Limit	Limit input (+) direction
/5	/11	/17	Home	Home input
/6	/12	/18	-Limit	Limit input (-) direction
20	26	32	Pulse	Pulse output (approx. 50% duty cycle)
21	27	33	Hold	Active output when motor stopped
/22	/28	/34	Input/External	INS signal (see Controller Status Port)
/23, /24	/29, /30	/35, /36	±Rampdown	When activated during a move in the (±) direction, respectively, the motor ramps down to starting velocity. Continues to run at starting velocity until deactivated or move complete.
25	31	37	Auxiliary Ground	Ground input optoisolator supply unless W2, W4, and W6 are jumpered to use PC Bus power (this is not recommended).

Note: A forward slash (/) preceding a signal name denotes an active LOW signal.

---

# 2

## Operation and Programming

## Theory of operation

The 5000 is intended for use with a pulse and direction input stepper motor driver. The 5000 is capable of a step rate of 240,000 pulses per second, which makes it ideal for microstepping applications.

Three independent intelligent stepper controllers produce profiled outputs optimized for each axis. All parameters can be changed *on the fly*, and the up and down ramps can be programmed independently. Each controller handles a full complement of limit inputs. The controllers can be operated in either a polled or an interrupt configuration. In a polled configuration, the status of each controller is read to determine when the motion is complete. In an interrupt configuration, each controller generates an interrupt when the end of the motion is complete. Complex profiles are possible by varying the profile parameters during operation.

To minimize the use of PC port space, the 5000 uses an indirect addressing scheme. Only two I/O addresses are occupied by the 5000 on the PC Bus. The location of these ports is determined by W7 (see Section 1) and the two rotary hex switches — SW2 and SW1. The address port is the even, or lowest port, while the data port is the odd, or highest.

To write to a port, you must first write to the indirect port address at the even I/O port and either read or write the odd (data) port. See Table 2-1 for port definitions.

Each controller occupies four indirect ports as indicated in the indirect address list. In addition, each controller has eight internal registers (R0 through R7). Table 2-2 describes the function of each register. Each register is accessed by first writing a Data Register Selection command to the command buffer of the respective controller. Register data are then written to the indirect address. The example below shows the steps necessary to access a typical register.

### Loading the Axis A counter

1. Write the indirect address of the Axis A command register (00h) to the board indirect address port (x00h).
2. Write the Data Register selection command for the counter register (80h) to the indirect data port (x01h).
3. Write the indirect address of Axis A register bits (0-7) buffer (01h) to the board indirect address port.
4. Write the least significant byte of the 24-bit counter value to the indirect data port.
5. Write the indirect address of Axis A register bits (8-15) buffer (02h) to the board indirect address port.
6. Write the next significant byte of the 24-bit counter value to the indirect data port.
7. Write the indirect address of Axis A register bits (16-23) buffer (03h) to the board indirect address port.
8. Write the most significant byte of the 24-bit counter value to the indirect data port.

While this procedure may seem complicated, you can reduce the number of steps by using the supplied driver functions on the companion software disk.

The status of each controller can be accessed by reading from the indicated indirect address using the bit assignments explained below. Additional status information is available by reading the state buffer at the indicated indirect address.

Register values can be calculated using the formula in paragraph *User-accessible registers*. If you have already installed the software into a working subdirectory, this is a trivial task. Use program PRO5000.EXE located in the subdirectory where you loaded the software utilities.



The operation of each controller is determined by the values loaded into its internal registers and the commands sent to the command register. For an explanation of these commands see paragraph *Command buffer*.

## Port locations

The 5000 uses indirect addressing. Two consecutive I/O addresses are used, and read and write access to different ports onboard the 5000 requires a write/read or a write/write sequence. This procedure may be simplified if the supplied drivers are used. Bus ports occupied by the 5000 are as follows:

A0	Port
0	Address
1	Data

### Addressing an onboard port

1. Write the address of the desired onboard port to the address port.
2. Write a byte to, or read a byte from the data port. For register assignments, see Table 2-1.

Table 2-1  
*Onboard port map*

Indirect Address	I/O Read	I/O Write
00h	Axis A Status	Axis A Command Buffer
01h	Axis A Counter (0-7)	Axis A Register (0-7)
02h	Axis A Counter (8-15)	Axis A Register (8-15)
03h	Axis A Counter (16-23)	Axis A Register (16-23)
04h-07h	Axis A Image	Axis A Images
08h	Axis B Status	Axis B Command Buffer
09h	Axis B Counter (0-7)	Axis B Register (0-7)
0Ah	Axis B Counter (8-15)	Axis B Register (8-15)
0Bh	Axis B Counter (16-23)	Axis B Register (16-23)
0Ch-0Fh	Axis B Image	Axis B Images
10h	Axis C Status	Axis C Command Buffer
11h	Axis C Counter (0-7)	Axis C Register (0-7)
12h	Axis C Counter (8-15)	Axis C Register (8-15)
13h	Axis C Counter (16-23)	Axis C Register (16-23)
14h-17h	Axis C Image	Axis C Images
18h	not used	Sync Latch (responds to all)
19h-1Fh	not used	Sync Latch Images
20h	not used	Reset Latch

Table 2-1  
Onboard port map (cont.)

Indirect Address	I/O Read	I/O Write
21h-27h	not used	Reset Latch Images
28h	PIC 1 A0 Read	PIC 1 A0 Write
29h	PIC 1 A1 Read	PIC 1 A1 Write
2Ah-2Fh	PIC 1 Images	PIC 1 Images
30h	Axis A State Buffer	not used
31h-37h	Axis A State Buffer Images	not used
38h	Axis B State Buffer	not used
39h-3Fh	Axis B State Buffer Images	not used
40h	Axis C State Buffer	not used
41h-47h	Axis C State Buffer Images	not used

Note: Write the desired register address through the address port, and read or write the data through the data port.

## Programming

This section explains how to access the controller registers, how to read the status port, how to simultaneously start the controllers, and how to execute a hard reset.

### Reading from an onboard port

1. Write the address from Table 2-1 for the desired onboard register to the address port.
2. Read the data from the data port.

### Writing the controller internal registers

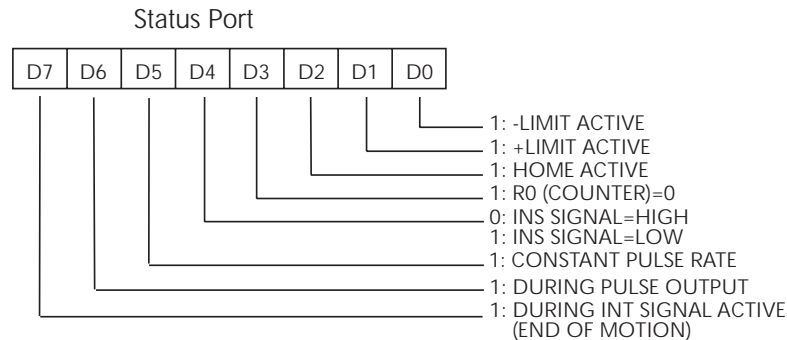
1. Write the address from Table 2-1 for the desired controller command buffer to the address port.
2. Write the register select command (8xh for the desired internal register to the data port (see data register selection in paragraph *Command buffer*).
3. Write the address from Table 2-1 for the register bits to be written for the selected controller to the address port.
4. Write the data to be loaded into the internal register to the data port.

### Writing to an onboard port

1. Write the address from Table 2-1 for the desired onboard register to the address port.
2. Write the data to be written to the data port.

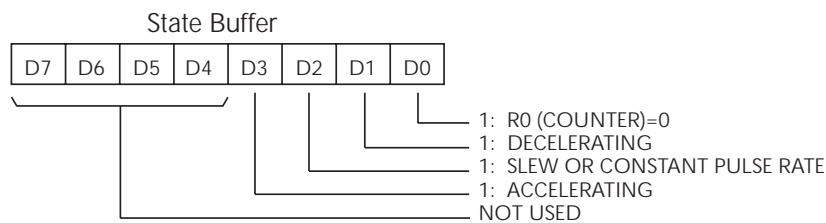
## Reading the stepper controller status port

Reading the controller status port gives the following information: (INT = interrupt output, INS = general purpose input signal).



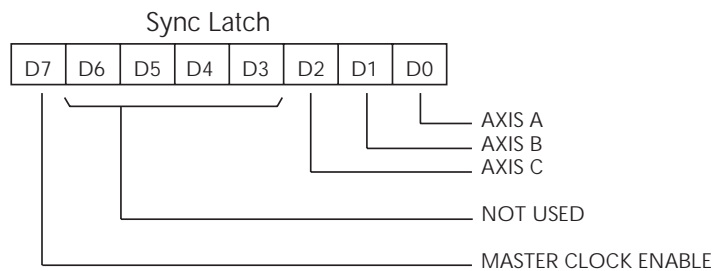
## Reading the state buffer

Reading the auxiliary status gives the following information:



## Sync latch

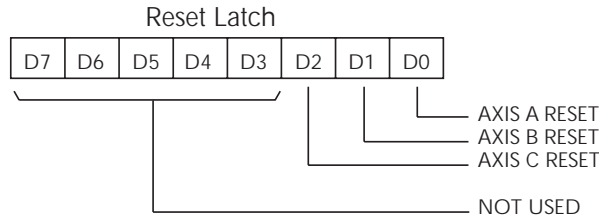
The Sync Latch provides a means of synchronizing the start of motion on all three axes at the same time. Under normal conditions, the motion starts as soon as the output mode selection command is issued. If you want to start all three axes simultaneously, Sync Latch can first be used to disable all three axes. All three controllers, or any combination of the three, can then be programmed for the desired motion and given the desired output mode selection command. Motion will not begin until the Sync Latch value is written to enable the desired controllers. By doing this, all three controllers can be started by writing a single bit. The Sync Latch is defined as follows:



For each bit of the Sync Latch, 0 = enabled, and 1 = disabled. Master Clock Enable (bit D7) must be LOW for any clock to operate. This enables a one-bit change to synchronize all controllers. On power-up, all clocks are enabled.

## Reset latch

This latch gives you the ability to hard reset any controller on the 5000. Individual bits are defined as follows:



For all bits, 0 = Reset, 1 = Enabled.

## Command buffer

Data bits — D6 and D7 — select 1 of 4 different command modes from the Command Buffer.

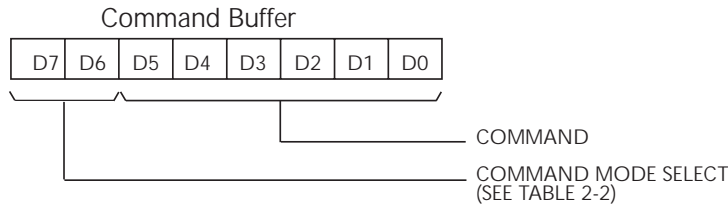


Table 2-2  
Command mode selection

D7	D6	Command Mode
0	0	Operating Mode
0	1	Control Mode
1	0	Data Register
1	1	Output Pulse Mode

Note: Command modes are selected from the command buffer.

## Operating mode selection

The operating mode initiates motion and selects the registers used to determine speed (FL, FH1, FH2), the type of profile (ramp or constant speed), and the motor stopping position.

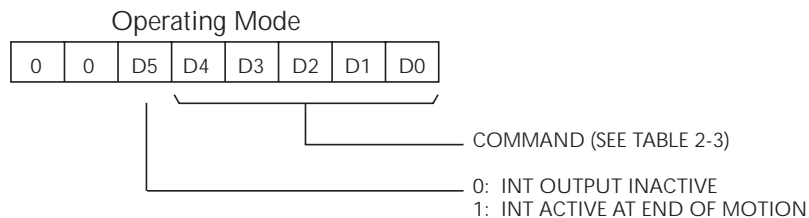


Table 2-3  
**Operating mode selection**

D4	D3	D2	D1	D0	Operating Mode Command
0	1	0	0	0	Soft Reset
1	0	0	0	0	Constant FL Speed†
1	0	0	0	1	Constant FH1 Speed†
1	0	0	1	1	Constant FH2 Speed†
1	0	1	0	1	High FH1 Speed†
1	0	1	1	1	High FH2 Speed†
1	0	1	0	0	Rampdown
1	1	0	0	0	Stop Immediately
1	1	1	1	1	Ramp Down and Stop

† Causes motion to begin.

Note: This mode initiates motions and selects the register used.

Bit D5 enables/disables INT which occurs at the end of motion. When D5 = 1, INT output will go active at the end of motion, and could cause PIC 0 to interrupt the CPU. When D5 = 0, INT output will remain inactive.

**Soft Reset.** This command must be issued at the beginning of every profile routine.

**Constant FL Speed.** This command will cause the controller to generate pulses at a constant rate determined by the value in the R1 Register. The other two constant speed commands (FH1 and FH2) operate identically with FH1 or R2 and FH2 or R3 registers to determine the pulse rate.

**High FH1/FH2 Speed Command Register.** The High FH1/FH2 speed command will cause the controller to generate pulses starting at the FL rate and will accelerate to the FH1/FH2 rate.

**Rampdown.** This command will cause the pulses out of the controller to decelerate from the FH1(R2) or FH2(R3) pulse rate (depending on which pulse rate command was active prior to the Ramp-Down Command) to the FL(R1) pulse rate.

**Stop Immediately.** This command will cause the pulses out of the controller to stop immediately.

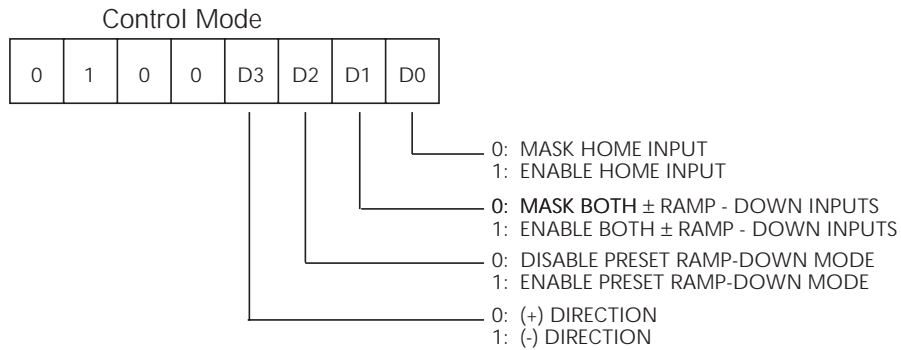
**Ramp Down and Stop.** This command will cause the pulses out of the controller to decelerate as described in the rampdown command, then once FL(R1) pulse rate is reached, the pulses out of the controller will stop.

## Control mode selection

Enables and disables Home,  $\pm$ Rampdown.

Enables and disables preset rampdown mode.

Sets the direction of motion.



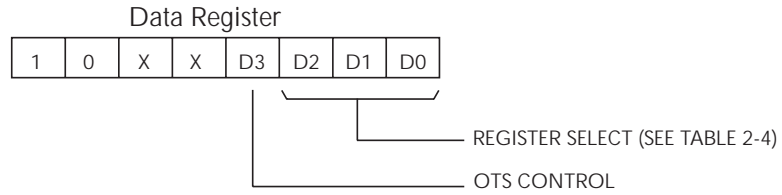
**Home.** When enabled ( $D0 = 1$ ), and when Home input goes LOW, the pulses from the pulse output will stop.

**$\pm$ Rampdown Inputs.** When enabled ( $D1=1$ ), pulses out of the controller pulse out line will decelerate to the constant FL ( $R1$ ) speed rate.

While operating in the constant  $xx$  speed mode, and with preset rampdown enabled, the pulses out of the controller will stop when the number of pulses out equals the value in register  $R0$ .

**Preset Rampdown.** While operating in the high  $xx$  speed mode, and with preset rampdown enabled, the pulse out of the controller will begin to decelerate when the number of pulses out equals the value in register  $R6$ . The pulse out rate will remain constant when the rate of the pulses equals the value in register FL or  $R1$ . It will stay this way until the number of pulses equals the value in register  $R0$ , in which case the pulses will stop.

## Data register selection



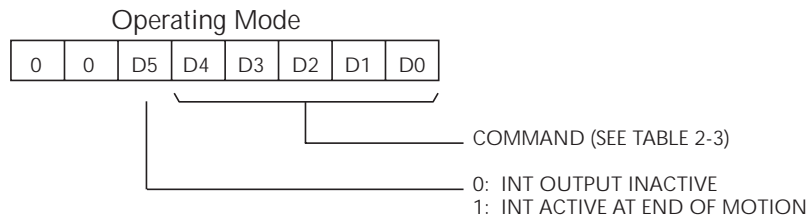
**OTS Control.** This is a general purpose output bit. To set this output low, write D3 = 1. To set this output HIGH, write D3 = 0.

Table 2-4  
Data register selection

D2	D1	D0	Data Register	Function
0	0	0	R0	Counter
0	0	1	R1	FL
0	1	0	R2	FH1
0	1	1	R3	FH2
1	0	0	R4	Acceleration Rate
1	0	1	R5	Deceleration Rate
1	1	0	R6	Rampdown Point
1	1	1	R7	Multiplier

Note: You can select any of 8 profile generation registers.

## Output mode selection



The output mode selects the output type, enabling of ramp up/down halfway, and enabling of the digital filter for 3 limit inputs.

Output type can be either pulse and direction type or +Pulse, -Pulse type.

When ramp up/down is disabled, the pulse-out will not accelerate or decelerate from its present pulse-out rate. This holds true even if a slow-down limit switch is activated.

For each axis, a digital filter can be activated for the  $\pm$ limits ( $\pm$ EL) and the Home inputs. When the filters are enabled, the input signals must be active LOW for at least four clock periods before the controller reacts to the inputs. This way noise, with short time periods, can not cause false triggering of the controller inputs.



## User-accessible registers

The profile of each stepper controller is controlled through eight user-accessible internal registers of varying lengths. See Table 2-5.

Table 2-5  
Controller register reference

Register	Function	Mode Used	Size (bits)	Read/Write
R0	Counter	High Speed Preset Constant Speed Preset	24	R/W
R1	FL	High Speed Preset Constant Speed Preset Constant Speed Cont Constant Speed Home Ret	13	W
R2	FH1	All	13	W
R3	FH2	All	13	W
R4	Accel	High Speed Preset High Speed Cont High Speed Home Ret	14	W
R5	Decel	High Speed Preset High Speed Cont High Speed Home Ret	14	W
R6	Rampdown point	High Speed Preset	20	W
R7	Multiplier	All	10	W

Note: The profile of each axis is controlled by 8 internal registers.

## Register descriptions

The registers listed below are used in all control modes.

**R0 - Down Counter: 24 Bits, Read/Write.** This register decrements with every pulse-out signal. When operating in the preset rampdown mode (see control mode selection in paragraph *Command buffer*), R0 must be loaded with the value of the total number of pulses for the entire profile routine. When the register decrements to zero, the controller will stop sending pulses.

When not operating in the preset rampdown mode, R0 will continue to decrement for every pulse-out but will not effect any operation. When it reaches a zero value, the next pulse-out will cause it to roll over to FFFFFFFh. The range of R0 is 000001h to FFFFFFFh (1 to 16,777,215d).

**R1 - Low Speed (FL) Register: 13 Bits, Write Only.** This register is loaded with a value that determines the pulse-out rate or frequency. Typically, this register is used to set the low operating frequency.

The frequency of a pulse-out is a function of the rate multiplier and the value of R1. See R7 below for further details in determining the rate multiplier value  $n$ . The range of R1 is 0001h to 1FFFh (1 to 8191d).

$$f_{\text{pout}} = nR1\text{Hz}$$

where  $f_{\text{pout}}$  is the pulse out frequency.

**R2 - FH1, High Speed 1 Register: 13 Bits, Write Only.** This register is loaded with a value that determines the pulse-out rate or frequency. Typically, this register is used to set one of two high operating frequencies. The range of R2 is 0001h to 1FFFh (1 to 8191d).

$$f_{\text{pout}} = nR2\text{Hz}$$

**R3 - FH2, High Speed 2 Register: 13 Bits, Write Only.** This register is loaded with a value that determines the pulse-out rate or frequency. The range of R3 is 0001h to 1FFFh (1 to 8191d).

$$f_{\text{pout}} = nR3\text{Hz}$$

**R4 - Acceleration Rate Register: 14 Bits, Write Only.** This register is loaded with a value that determines the acceleration rate of the pulse-out signal. Any time a command causes an acceleration, the rate will always be determined by the acceleration rate register.

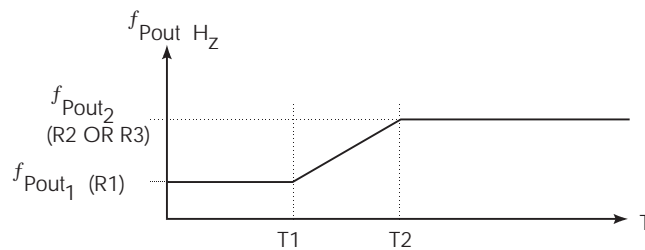
$$\text{Acceleration Rate} = \frac{f_{\text{pout}2} - f_{\text{pout}1}}{t_2 - t_1}$$

The acceleration rate is a function of the value in R4, the rate multiplier  $n$ , and  $f_{\text{clock}}$ .

$$\text{Acceleration Rate} = \frac{nf_{\text{clock}}}{R4} \text{ pulses per second}^2$$

The range of R4 is 0002h to 3FFFh (2 to 16,383d).

Figure 2-1  
Acceleration profile



**R5 - Deceleration Rate Register: 14 Bits, Write Only.** This register is loaded with a value that determines the deceleration rate of the pulse-out signal. Any time a command or a limit switch,  $\pm$  rampdown causes a deceleration, the rate of deceleration will always be determined by this register. The range of R5 is 0002h to 3FFFh (2 to 16,383d).

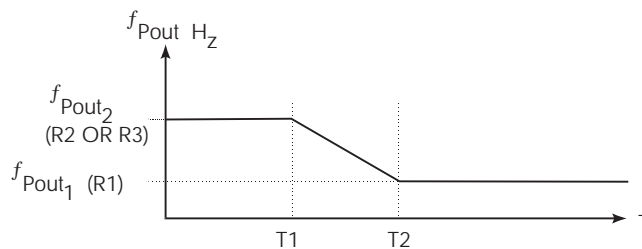
$$\text{Deceleration Rate} = \frac{f_{\text{pout2}} - f_{\text{pout1}}}{t_2 - t_1}$$

The deceleration rate is a function of the value in R5, the rate multiplier n, and  $f_{\text{clock}}$ .

$$\text{Deceleration Rate} = \frac{nf_{\text{clock}}}{R5} \text{ pulses per second}^2$$

Figure 2-2

**Deceleration profile**



**R6 - Rampdown Point Register: 20 Bits, Write Only.** The rampdown point register is used internally in the controller while operating only in the preset rampdown mode. When the value in register R6 is greater than the value in R0 (down counter), the pulse-out rate will begin to decelerate at the rate determined by the deceleration rate register. Deceleration will occur only if the pulse-out rate is within the high speed rate determined by register R2(FH1) or R3(FH2). Once the pulse-out rate is equal to the low speed rate, deceleration will stop and the pulse-out rate will remain at the low speed rate until the down counter equals zero, in which case pulse-out will stop. When using the Rampdown Point Register (while in preset rampdown mode), it is desirable to set the rampdown point so that the count-down register equals a small value once the low speed rate is reached. (This is sometimes referred to as a *porch*.) The formula below determines the value of R6 for a given profile so that the count-down register will have a value of 10 when the Low Speed Rate is reached during the deceleration.

$$R6 = \frac{R5[(R2^2 \text{ or } R3^2) - R1^2]}{16384 R7} + 10$$

The range of R6 is 00001h to FFFFFh (1 to 1,048,575d).

**R7 - Rate Multiplier Register: 10 Bits, Write Only.** This register is loaded with a value that determines the multiplier value of the speed registers R1, R2, and R3. The rate multiplication factor  $n$  which is typically set to 1.0 is given as:

$$n = \frac{f_{\text{clock}}}{8192 R7}$$

The range of R7 is 002h to 3FFh (*2 to 1023d*).

If:

$$R7 = \frac{f_{\text{clock}}}{8192}$$

then  $n$  is equal to 1.0, and the values loaded into R1, R2, and R3 are in pulses per second. In addition, because  $n = 1$ , many of the formulas are simplified.

---

# 3

## Interrupt Control

## Description of interrupt control

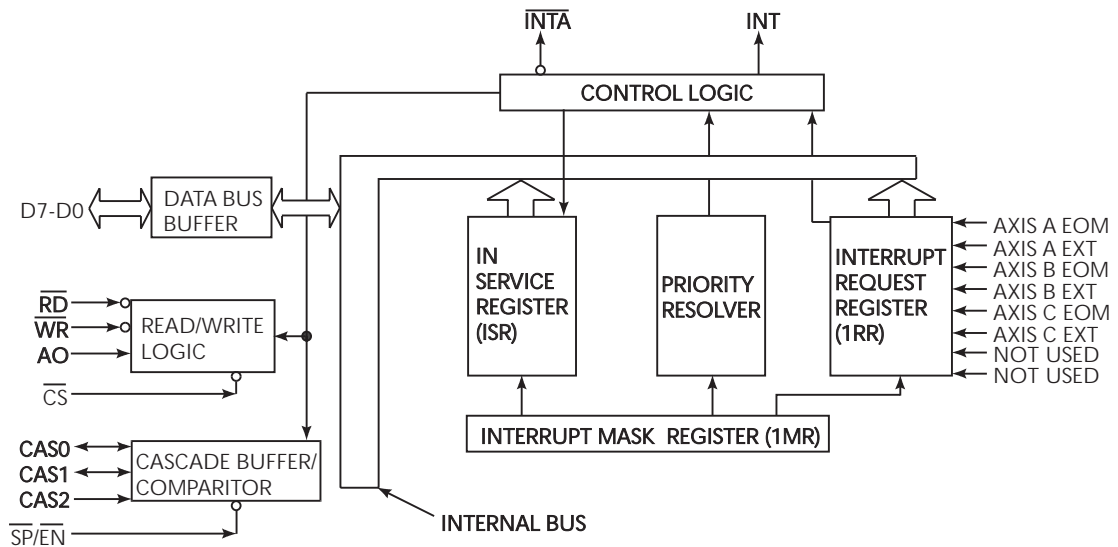
The 5000 uses an 8259A Programmable Interrupt Controller (PIC) to handle the interrupt sources on the board. Each axis can generate two interrupts. The first is an end-of-motion (EOM) issued when a programmed move is completed. The second interrupt is generated from INS, a user-defined signal available at J1 (22, 28, and 34) for axes A, B, and C, respectively.

The PIC may only be used in a nonvectored mode (polled). When a board-level interrupt is generated, the PIC must be polled to determine which interrupt was triggered. The PIC may be disabled by removing jumper W8.

Only the features of the 8259A PIC that are used by 5000 are discussed below. For complete information on the 8259A PIC refer to Intel's *Peripheral Components Manual*.

Figure 3-1 shows a functional block diagram of the 8259A PIC. On the 5000, the PIC is connected directly to the PC Bus through jumper W8.

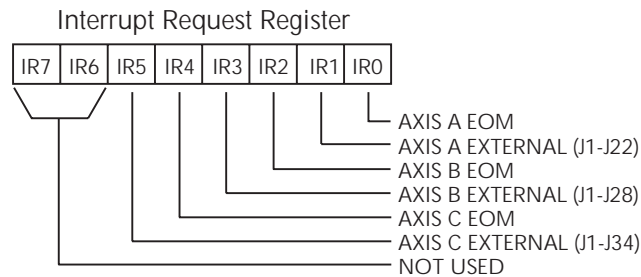
Figure 3-1  
PIC block level diagram



Note: The 5000 uses an Intel 8259A and is directly connected to the PC Bus through jumper W8.

## Interrupt Request Rgtr (IRR), In-Service Rgtr (ISR)

The interrupts at the interrupt request input lines (IR0-IR5) are handled by the IRR and the ISR. The IRR is used to store the interrupts requesting service, and the ISR is used to store the interrupts being serviced. Interrupts and IRR/ISR bits correspond to the format below.



## Priority Resolver (PR)

The Priority Resolver (PR) block determines the priority of the bits set in the IRR. The highest priority bit is selected and strobed into the corresponding ISR bit at the time of the poll command.

## Interrupt Mask Register (IMR)

The Interrupt Mask Register stores the bits that determine the interrupt lines to be masked. The IMR operates on the IRR. Masking a higher priority input will not affect the interrupt request lines of lower priority. Masking disables the interrupt for the masked input.

## Interrupt Output (INT)

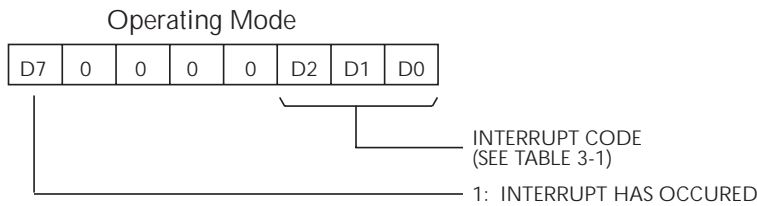
The Interrupt Output (INT) signal indicates that the PIC has an interrupt request pending. This signal can be routed to PC Bus interrupt IRQ2 through IRQ7 (IRQ2 default) via W8. The poll command causes interrupt status to be placed on the bus during the next read of the PIC.

# PIC operation

## Interrupt sequence, 80x86/80x88 mode

The sequence of events during an interrupt when using an 80x86/80x88 CPU is as follows:

1. One or more of the interrupt request lines (IR0-IR7) are raised high setting the corresponding IRR bit(s).
2. The PIC evaluates these requests and sends an interrupt request to the CPU provided that jumper W8 is installed.
3. The interrupt is acknowledged by your program interrupt service routine by writing a poll command (OCW3) to the PIC.
4. The CPU reads the PIC to obtain the priority level as shown below. After the read, the high ISR bit is set, and the corresponding IRR bit is reset.



*Table 3-1  
Interrupt code*

D2	D1	D0	Interrupt Request
0	0	0	IRQ0
0	0	1	IRQ1
0	1	0	IRQ2
0	1	1	IRQ3
1	0	0	IRQ4
1	0	1	IRQ5
1	1	0	IRQ6
1	1	1	IRQ7

Note: The interrupt code returns the highest priority interrupt and sets the corresponding ISR bit.

5. The previous step completes the interrupt cycle. In the Automatic End-Of-Interrupt (AEOI) mode, the ISR bit is reset on the read following the poll command. Otherwise, the ISR bit remains set until an appropriate End-Of-Interrupt (EOI) command is issued.

## End-of-Interrupt command

After the priority level is read, the ISR bit must be reset. This is done with EOI command from the host PC, or it can be done automatically in the AEOI mode. There are two forms of the EOI command — Specific and Non-Specific. A Non-Specific EOI command resets the highest ISR bit of those that were set, and a Specific EOI command can be issued to reset a specified ISR bit.



## Completing an interrupt

You have to provide an interrupt routine to trap IR7 interrupt glitches that appear as interrupt 7 requests. In the AEOI mode, the ISR bit for the interrupt being serviced is reset automatically at the interrupt return in your interrupt routine.

## Operating modes

### Fully nested mode

This is the default mode entered after initialization unless another mode is programmed. In this mode, interrupt requests are ordered in priority from 0 through 7 with 0 being the highest priority. When a poll command is received and the priority level is read, the highest priority request is placed on the data bus. The corresponding bit in the ISR is also set. It stays set until the CPU issues an EOI command, or, if in AEOI mode, until the priority level is read. While the ISR bit is set, all further interrupts of equal or lower priority are inhibited. Interrupts of higher priority will issue an interrupt request (which will be acknowledged only if the PC has unmasked the interrupt).

### Special mask mode

This mode is similar to the fully nested mode except that when a bit in the ISR is set, it only inhibits interrupt requests at that level. All other unmasked interrupt requests (lower as well as higher) are enabled.

### Specific rotation (specific priority)

The default priority of interrupts is IR0 (highest) through IR7 (lowest). This can be changed using the set priority command. This command specifies one input as having the lowest priority and fixing all other priorities. For example, if IR2 is specified as having the lowest priority, the priority of interrupts will be: IR3 (highest), IR4, IR5, IR6, IR7, IR0, IR1, IR2 (lowest).

### Automatic rotation (equal priority)

In this mode, a device, after being serviced, receives the lowest priority. Such a device requesting an interrupt would have to wait until all other devices have been serviced.

### Non-vectored mode (poll command)

The PIC must be polled for interrupt status. To do this, the poll command is written to the PIC, and then the status is read. The PIC treats the read pulse as an INTA pulse. The interrupt is frozen from the write to the read.

## PIC programming

The PIC accepts two types of command words from the CPU — Initialization Command Words (ICW) and Operational Command Words (OCW).

### Initialization Command Words (ICW)

Before normal operation can begin, the PIC must be brought to a starting point by a sequence of three bytes. Figure 3-2 shows the initialization sequence. Whenever a command is written to the PIC low port with bit D4 = 1, it is interpreted as ICW1. Register ICW1 starts the initialization sequence during which the following automatically occur:

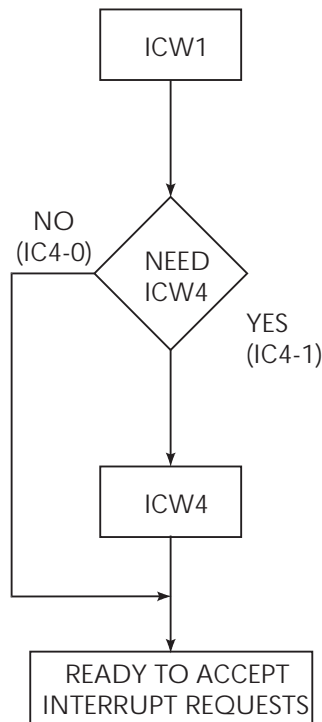
1. The IMR is cleared.
2. IR7 is assigned the lowest priority.
3. The slave mode address is set to 7.
4. Special Mask Mode is cleared.
5. Status Read is set to IRR.
6. If IC4 = 0, then all functions selected in ICW4 are set.

ICW2 is not used by the 5000. However, a value must be written to it to properly initialize the PIC. Any value may be written since the value will have no effect on PIC operation.

ICW3 is not used by the 5000.

Writing to the ICW4 completes the initialization sequence.

Figure 3-2  
**PIC initialization sequence**



Note: Bit D4 set assumes the next word issued will be ICW4.

## ICW1 format and description

Figure 3-3 shows the format for ICW1 and ICW4. Set the bits for the 5000 to the PIC low port in the following manner:

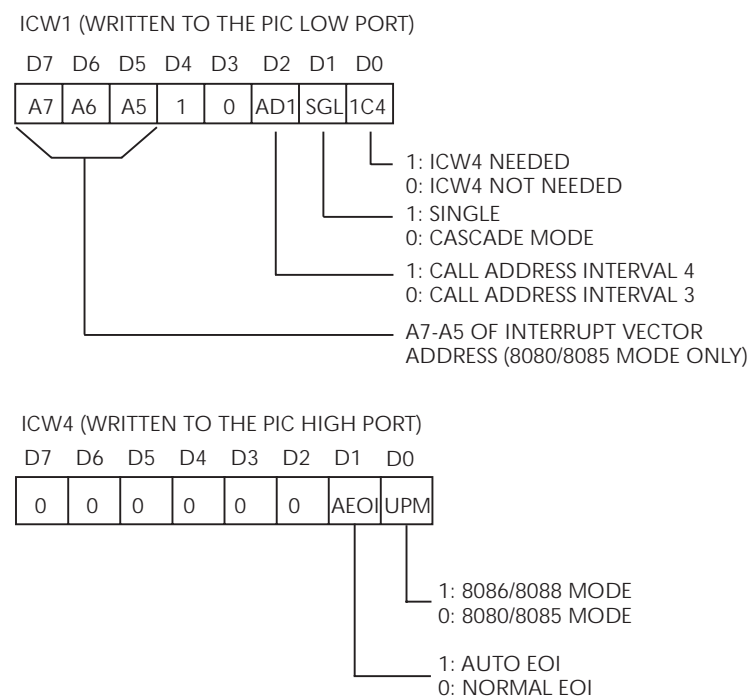
**D5-D7 (A7-A5):** May be set or cleared. This bit has no impact on the operation of the 5000.

**D2 (ADI):** May be set or cleared. This bit has no impact on the operation of the 5000.

**D1 (SNGL):** Set this bit.

**D0 (IC4):** If ICW4 is needed, set this bit. ICW4 is needed if the CPU is an 80x86/80x88 or if AEOI mode is desired.

Figure 3-3  
**PIC ICW format**



Note: Write ICW1 to the PIC low port and ICW4 to the PIC high port depending on how you initialize.

## ICW4 format and description

ICW4, written to the PIC high port, is read only if D4 of ICW1 is set. Set the data bits in the following manner:

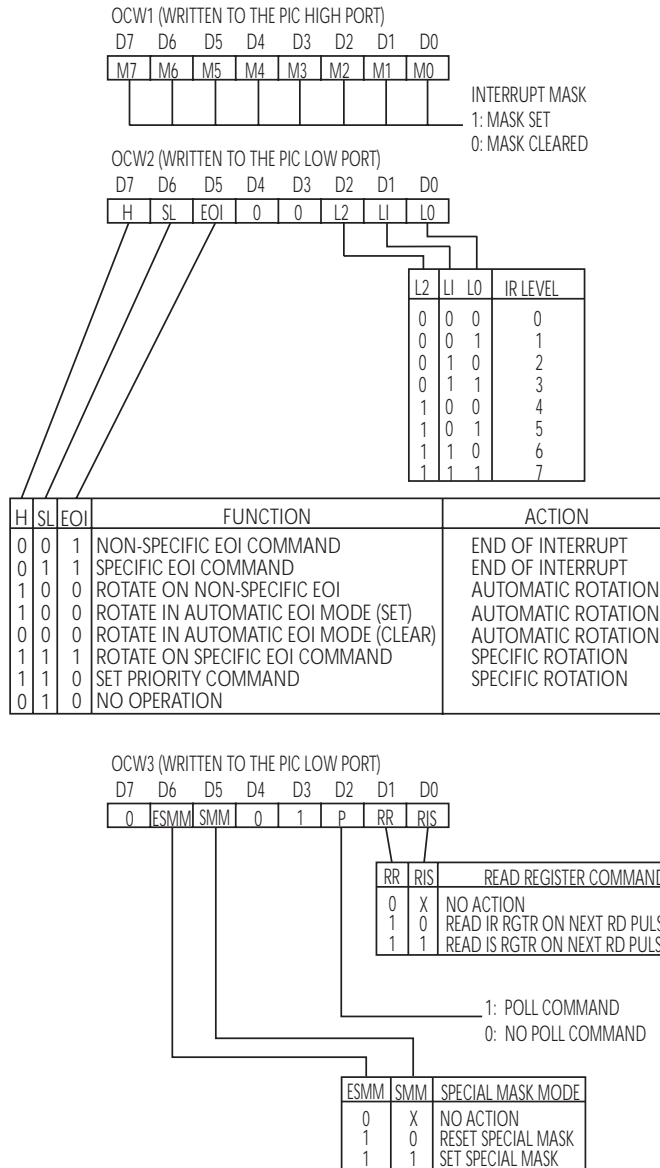
**D0 (UPM):** Set this bit for operation in 8086 or 8088 mode. Clear this bit for operation in 8080 or 8085 mode.

**D1 (AEOI):** Set this bit for AEOI mode, and clear it for normal EOI.

## Operation Command Words (OCW)

These are the words that command the PIC to operate in various interrupt modes. The OCW can be written to the PIC anytime after initialization. Figure 3-4 shows the format for OCW.

Figure 3-4  
PIC OCW format



Note: Write to the OCW at any time after initialization.

## OCW1 format and description

Written to the PIC high port, OCW1 sets and clears the mask bits in the Interrupt Mask Register (IMR). M7-M0 represent the eight mask bits for IR7-IR0, respectively. M = 1 means that the input is masked (inhibited), while M = 0 means that the input is enabled. Since IR6 and IR7 are not used by the PIC, set M6 and M7. Reading OCW1 through the PIC high port returns the interrupts that are masked.

## OCW2 format and description

The bits in OCW2, written to the PIC low port, are defined as follows:

**D7 (R):** Used to control all PIC rotation operations. If R is set, a form of priority rotation will be executed depending on the operation selected.

**D6 (SL):** Used to select a specific level for a given operation. If set, L0-L2 are enabled, and the operation selected will be executed on the specified interrupt level.

**D5 (EOI):** Used for all EOI commands (except AEIOI). If set, a form of EOI will be executed.

**D0-D2 (L0-L2):** Designates an interrupt level (0-7) to be acted upon for the operation selected by the EOI, SL, and R bits. L0-L2 are enabled or disabled by the SL bit.

All the possible operations by OCW2 are shown in Figure 3-4. A brief description of each is given below.

## OCW2 commands

**Non-Specific EOI Command:** Of the ISR bits that are set, the one with the highest priority is cleared.

**Specific EOI Command:** The ISR bit specified by LO-L2 is cleared.

**Rotate on Non-Specific EOI Command:** Same as Non-Specific EOI, except that when an ISR bit is cleared, its corresponding IR is assigned the lowest priority.

**Rotate in AEIOI:** Same as Rotate on Non-Specific EOI command, except that priority rotation is done automatically after the last INTA pulse. Setting the bit enters this mode, and clearing the bit exits this mode.

**Rotate on Specific EOI:** Same as Specific EOI, except that after the specified ISR bit is cleared, its corresponding IR is assigned the lowest priority.

**Set Priority Command:** The specified bit is assigned the lowest priority.

## OCW3 format and description

The bits in OCW3, written to the PIC low port, are defined according to the following:

**D6 (ESMM):** Enable Special Mask Mode. When set, it enables the SMM bit (see below) to set or reset the Special Mask Mode. When ESMM is cleared, the SMM bit becomes a don't care.

**D5 (SMM):** Special Mask Mode. If ESMM and SMM are set, the PIC will enter the Special Mask mode. If ESMM is set and SMM is cleared, then, the PIC will revert to normal mask mode.

ESMM	SMM	Mode
1	1	Special Mask Mode
1	0	Normal Mask Mode

**D2 (P):** Polled Mode. If set, the next read of the PIC low port will return the highest priority level requesting the interrupt if an interrupt has occurred. See Figure 3-3.

**D1 (RR):** Read Register. If set, the next read of the PIC low port will return IIR and ISR status, depending on the RIS bit (see below). If RR is cleared, the RIS bit becomes a don't care.

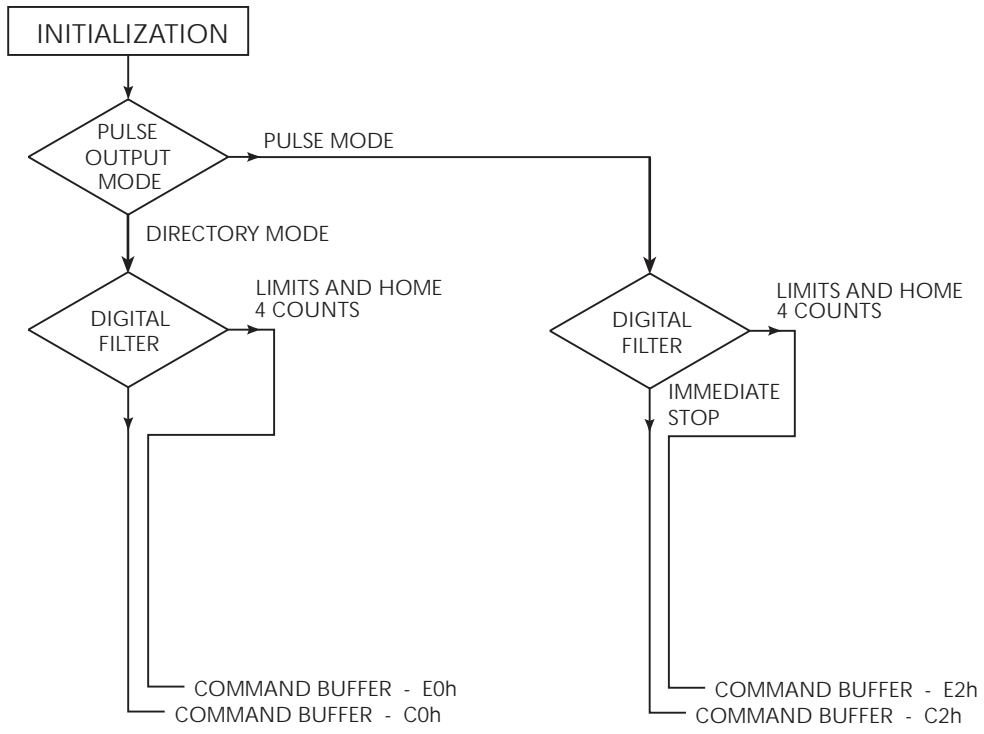
**D0 (RIS):** If P = 0, RR = 1, and RIS = 0, the next read of the PIC low port will return the IRR status. If P = 0, RR = 1, and RIS = 1, the next read of the PIC low port will return the ISR status.

P	RR	RIS	Next Read of Low Port
0	1	0	Return IIR Status
0	1	1	Return ISR Status

---

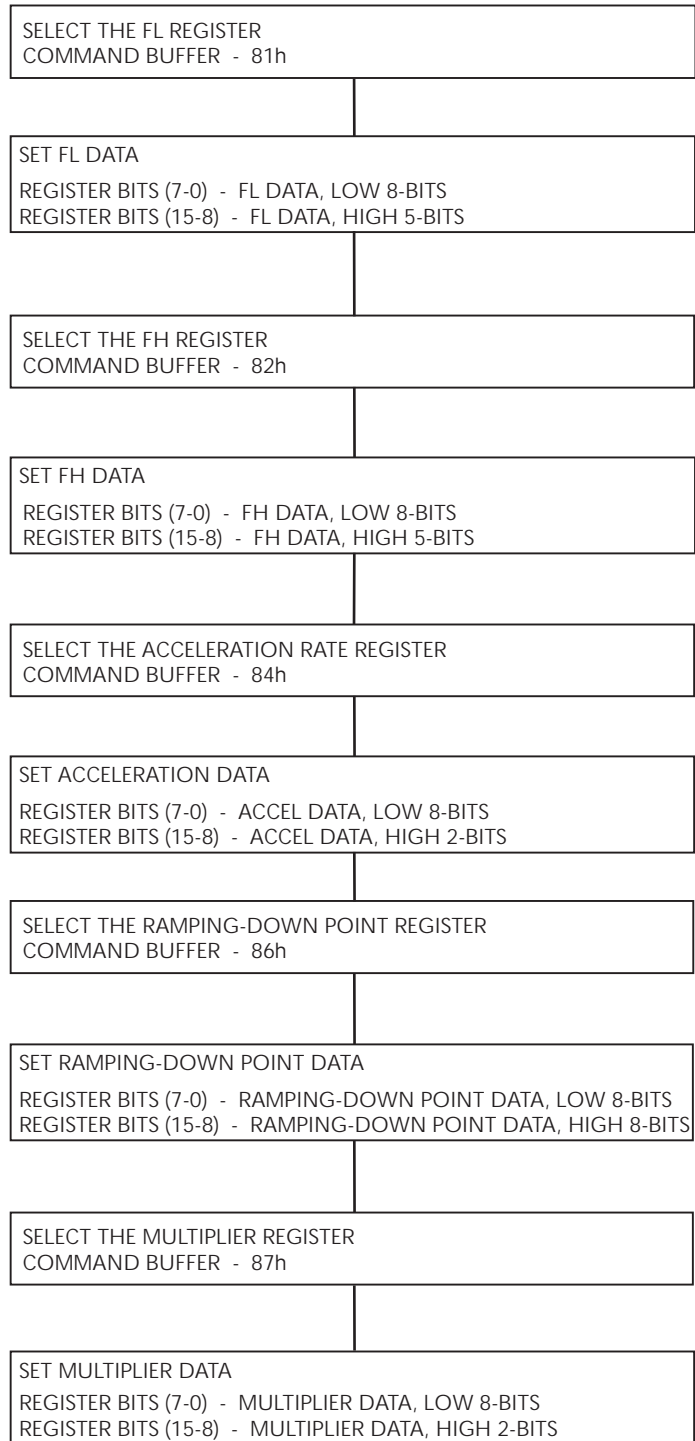
# **A** Typical Operation Procedures

# Initialization flow chart

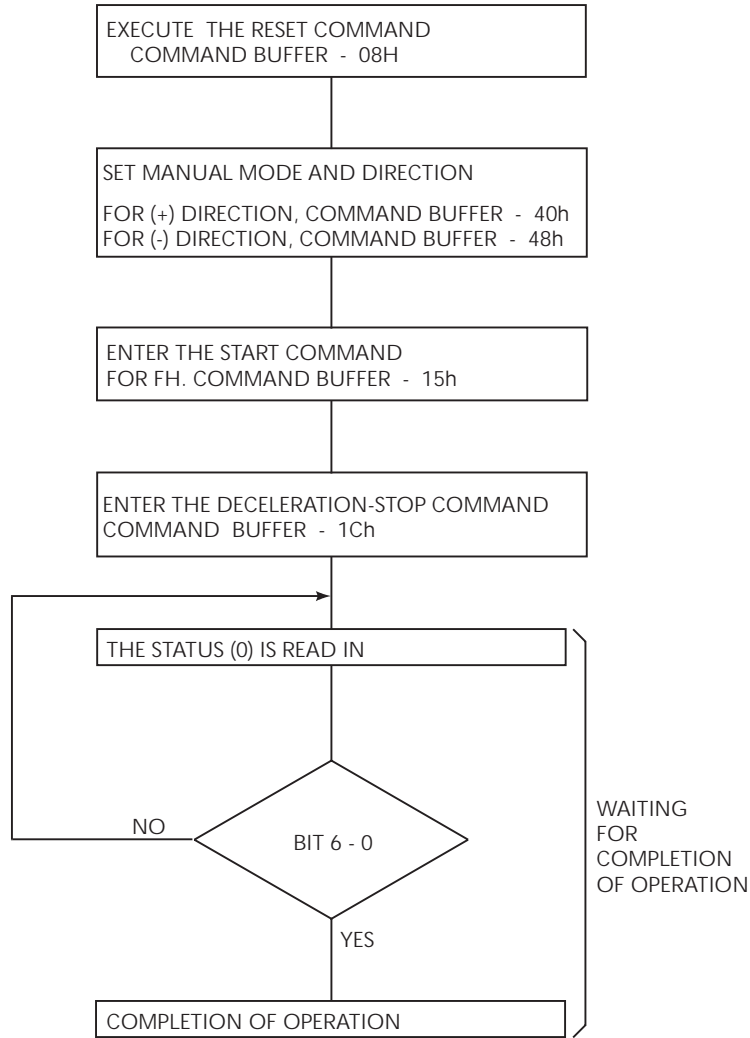
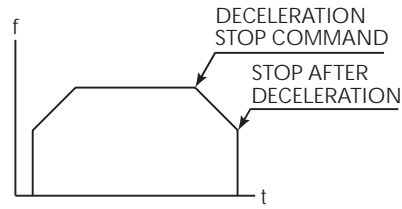




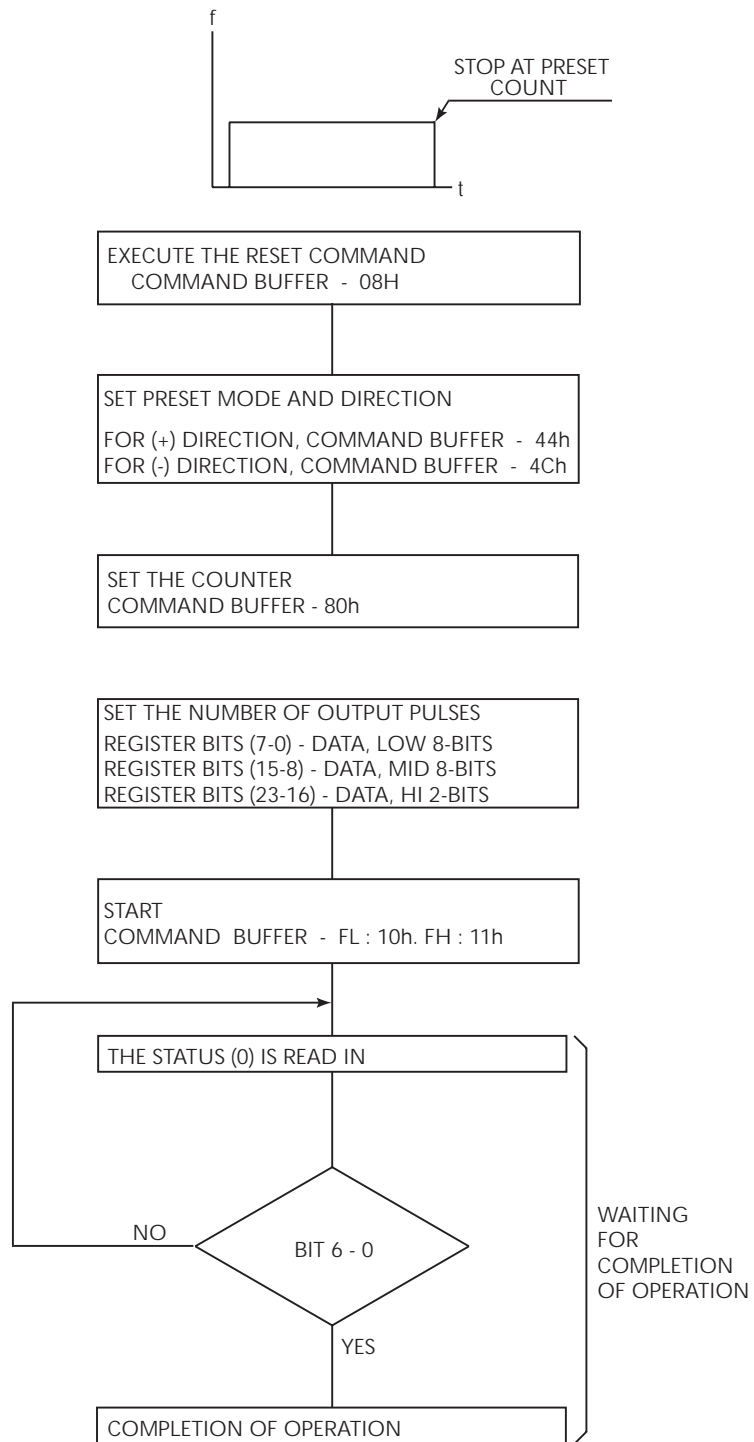
## Setting speed data



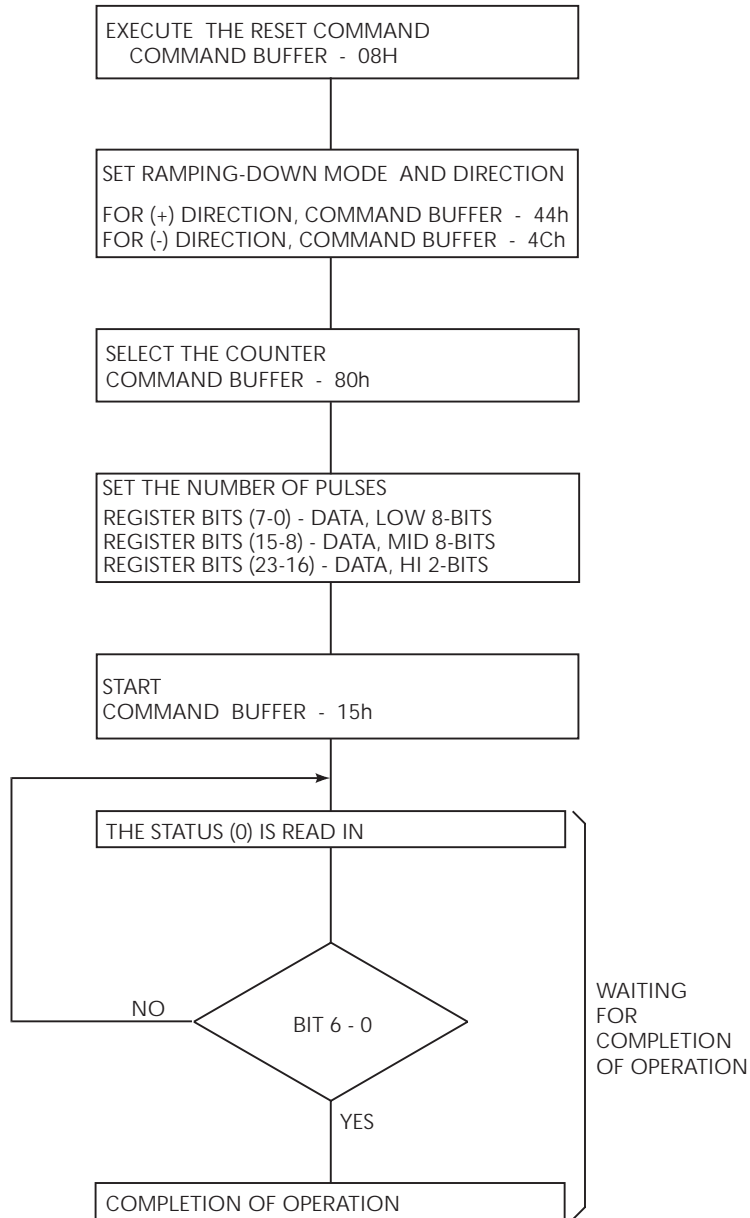
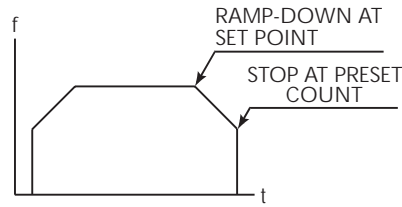
# High speed preset mode



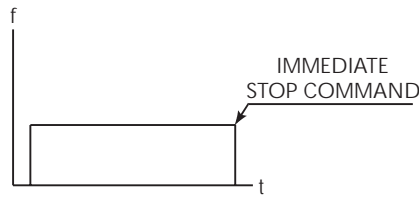
# Constant speed preset mode



# High speed continuous mode



# Constant speed continuous mode



EXECUTE THE RESET COMMAND  
COMMAND BUFFER - 08H

SET MANUAL MODE AND DIRECTION  
FOR (+) DIRECTION, COMMAND BUFFER - 40h  
FOR (-) DIRECTION, COMMAND BUFFER - 48h

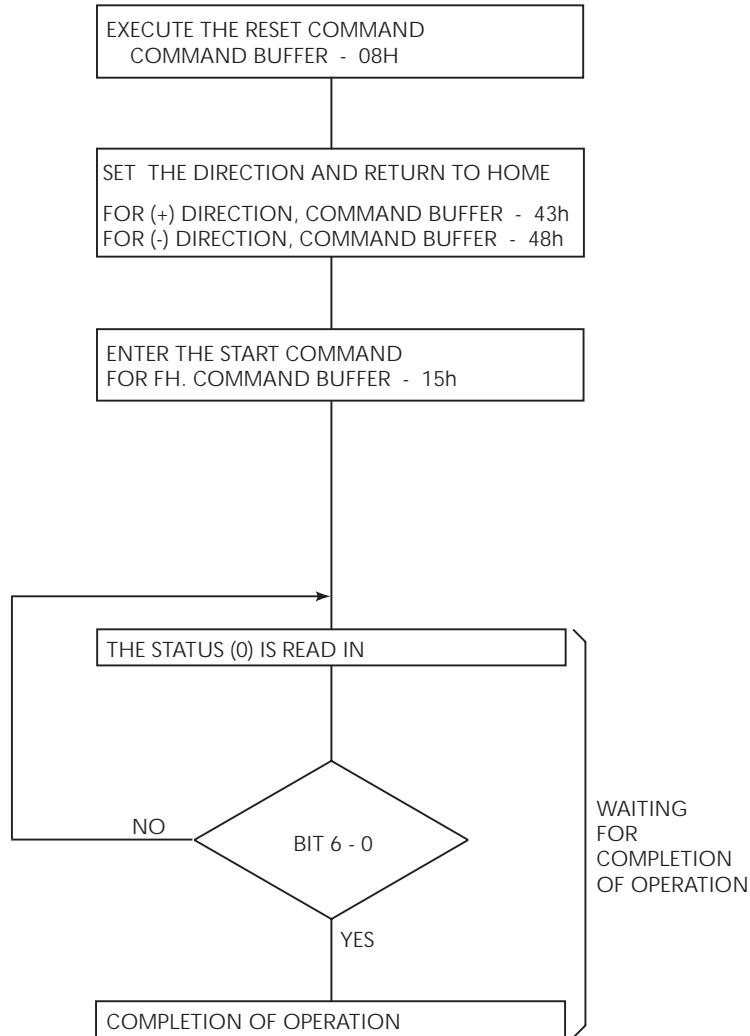
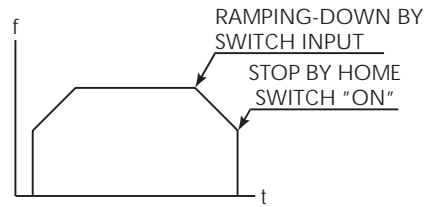
ENTER THE START COMMAND  
COMMAND BUFFER - FL : 10h. FH : 11h

.....

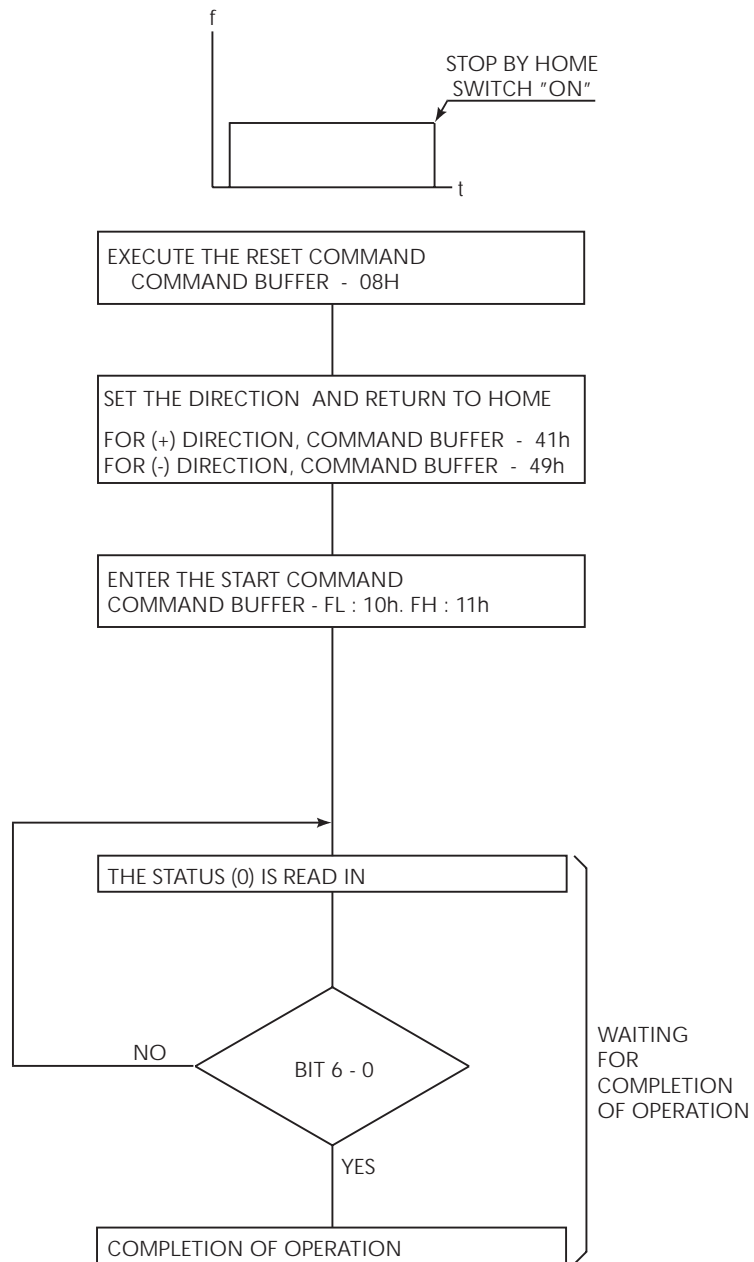
ENTER THE IMMEDIATE STOP COMMAND  
COMMAND BUFFER - 08h (RESET COMMAND)

COMPLETION OF OPERATION

# High speed return to home

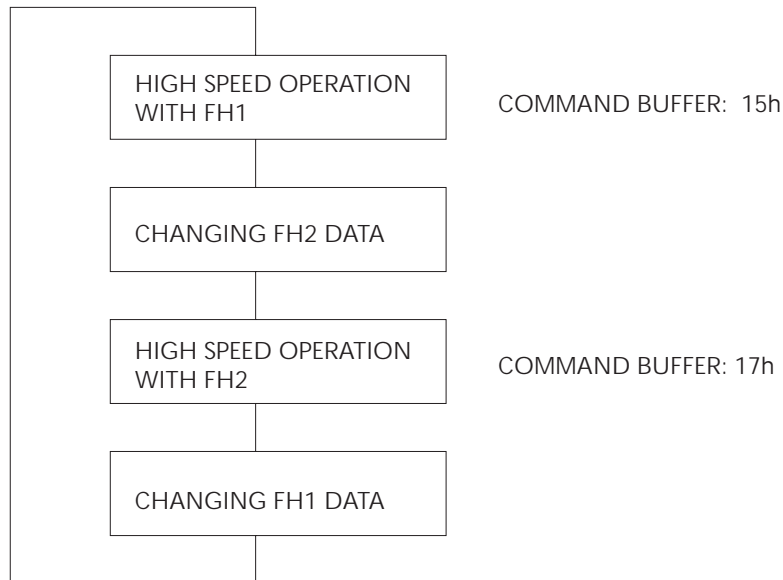


# Constant speed return to home



## Speed change during operation

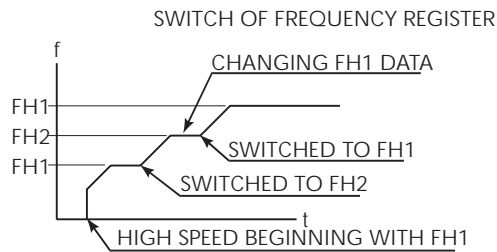
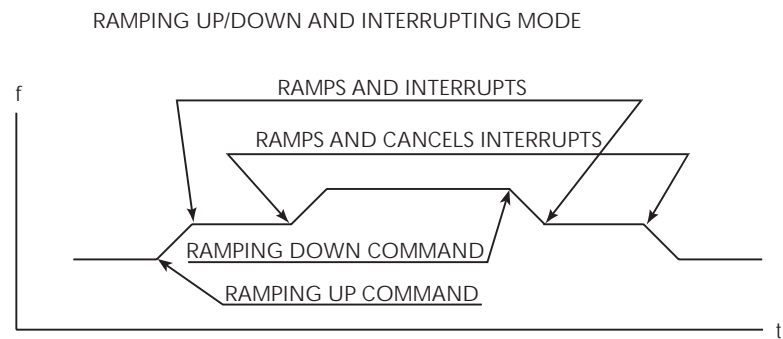
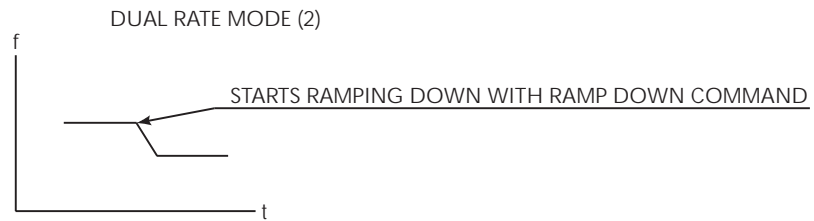
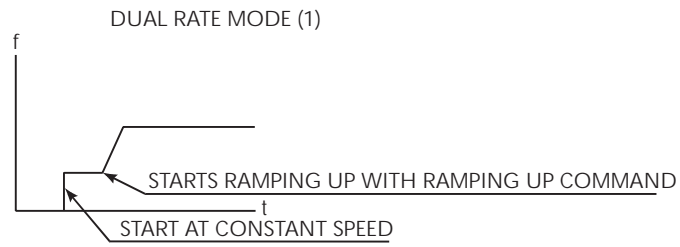
The 5000 allows the operator to change speeds during operation. By entering an appropriate operating mode selection command involving registers FH1 or FH2, the operator can cause the pulse output to ramp up or down to a rate different from the current rate. The contents of FH1 or FH2 may be changed during an operation without influencing that operation and later used to change speeds. The velocity rate may be changed on the fly an unlimited number of times.



To let the pulse-out ramp up or down to another rate after starting in the constant speed mode, change D2 of the start command to a '1'. Enter the command before changing rate.



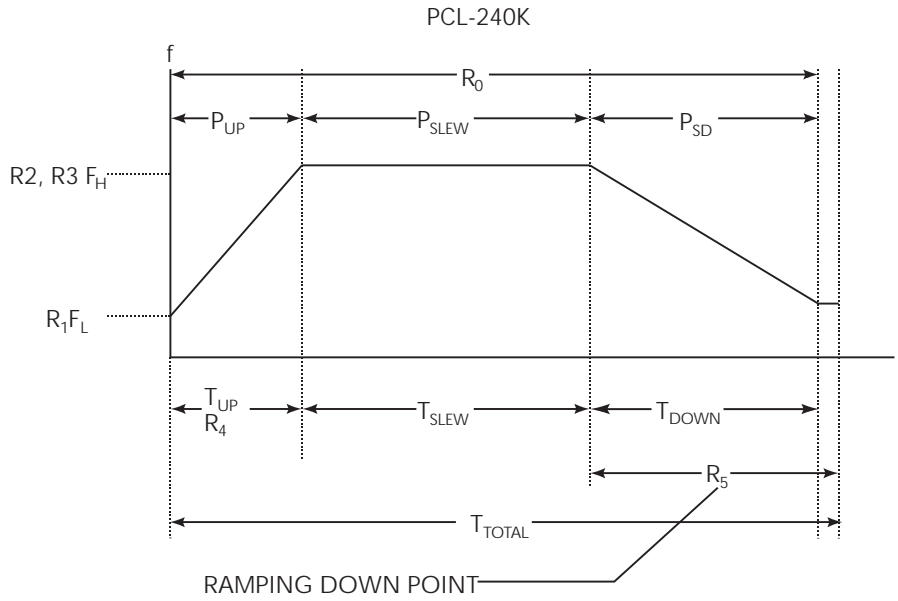
# Additional operating parameters



---

# **B** Typical High Speed Preset Mode Calculations

# Typical high speed preset mode calculations



$$FL, FH1, FH2 = (R1, R2, R3)n = (R1, R2, R3)\left(\frac{f_{clock}}{8192R7}\right)pps$$

$$T_{up} = \frac{(R2, R3 - R1)R4}{f_{clock}} \text{seconds}$$

$$tT_{down} = \frac{(R2, R3)^2 - R1^2}{f_{clock}} R4 \text{seconds}$$

$$P_{sd} = \frac{((R2, R3)^2 - R1^2)R4}{16384R7} \text{pulses}$$

$$P_{FL} = \text{Number of pulses after rampdown complete (porch)}$$

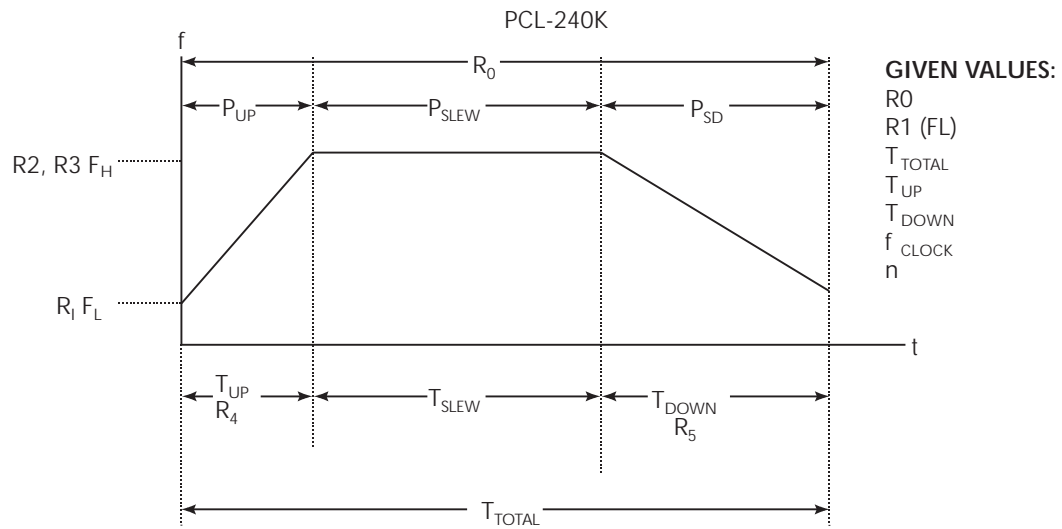
$$P_{up} = \frac{((R2, R3)^2 - R1^2)R5}{16384R7} \text{pulses}$$

$$P_{slew} = T_{slew} R2, R3 \text{ n pulses}$$

$$n = \frac{f_{clock}}{8192R7}$$

$$\text{Accel, Decel} = \frac{f_{clock}}{R4} pps^2$$

### Determine R2 to R7 from given values



$$R2, R3 = \frac{2R0 - nR1(T_{total} - T_{slew})}{n(T_{total} + T_{slew})}$$

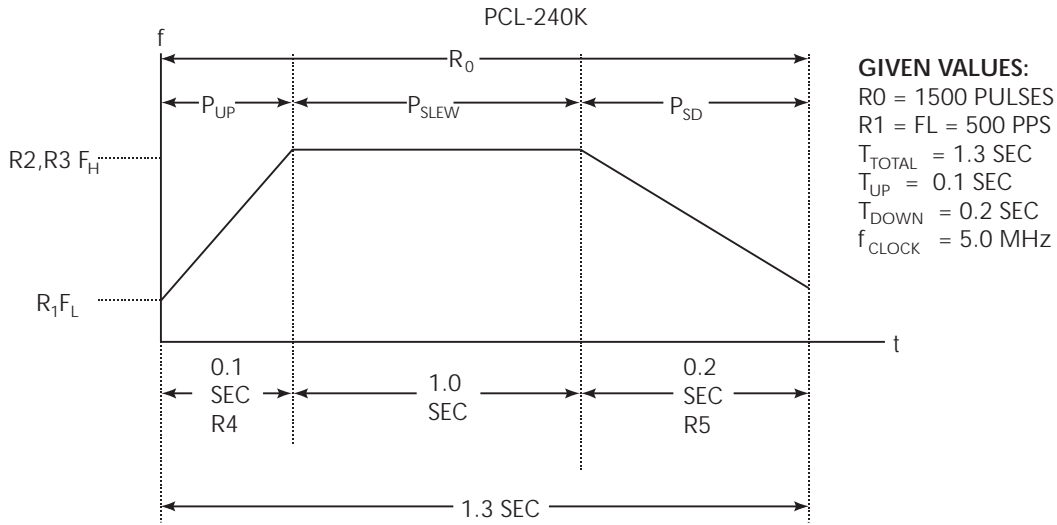
$$R4 = T_{up} \left( \frac{f_{clock}}{R2 - R1} \right)$$

$$R5 = T_{down} \left( \frac{f_{clock}}{R2 - R1} \right)$$

$$R6 = \left( \frac{R5}{R7} \right) \left( \frac{R2^2 - R1^2}{16384} \right)$$

$$R7 = \frac{f_{clock}}{8192n}$$

### Example 1



$$R2, R3 = \frac{2R0 - nR1(T_{total} - T_{slew})}{n(T_{total} + T_{slew})} = \frac{(2)(1500) - (1.0)(500)(1.3 - 1.0)}{(1.0)(1.3 + 1.0)} = 1239 = FH1, FH2$$

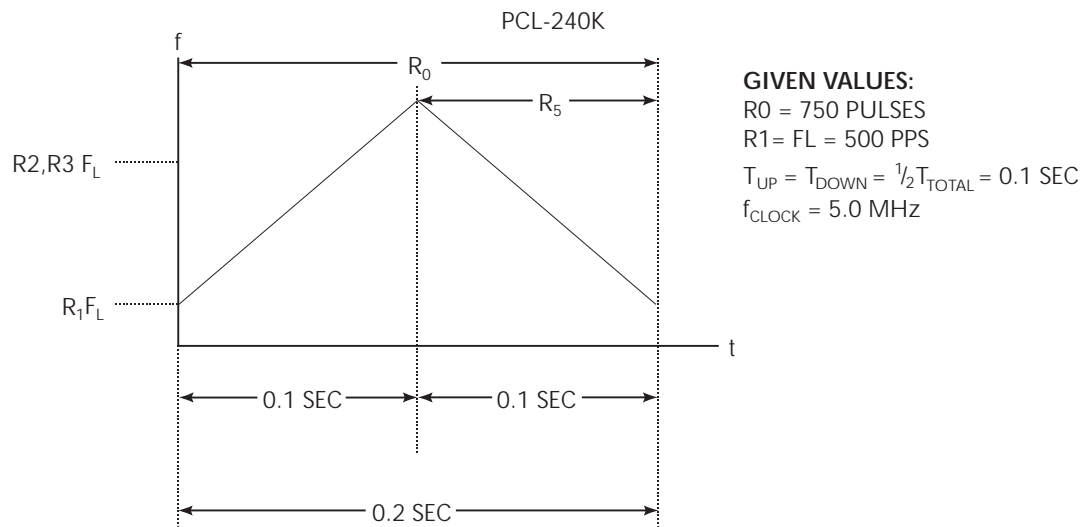
$$R4 = T_{up} \left( \frac{f_{clock}}{(R2, R3) - R1} \right) = (0.1) \left( \frac{5.0 \times 10^6}{1239 - 500} \right) = 677$$

$$R5 = T_{down} \left( \frac{f_{clock}}{(R2, R3) - R1} \right) = (0.2) \left( \frac{5.0 \times 10^6}{1239 - 500} \right) = 1353$$

$$R7 = \frac{f_{clock}}{8192n} = \frac{5.0 \times 10^6}{(8192)(1.0)} = 610$$

$$R6 = \left( \frac{R5}{R7} \right) \left( \frac{(R2, R3)^2 - R1^2}{16384} \right) = \left( \frac{1353}{610} \right) \left( \frac{1535121}{610} \right) = 174 = P_{sd}$$

### Example 2



$$R_2, R_3 = \left( \frac{2R_0}{nT_{total}} \right) - R_1 = \left( \frac{(2)(750)}{(1.0)(0.2)} \right) - 500 = 7000 = FH1, FH2$$

$$R_4 = T_{up} \left( \frac{f_{clock}}{R_2, R_3 - R_1} \right) = (0.1) \left( \frac{5.0 \times 10^6}{7000 - 500} \right) = 77$$

$$R_5 = R_4 = 77$$

$$R_6 = \frac{R_0}{2} = 375 = P_{sd} = P_{up}$$

$$R_7 = \frac{f_{clock}}{8192n} = \frac{5.0 \times 10^6}{(8192)(1.0)} = 610$$

---

# C

## PC I/O and Interrupt Mapping

## PC I/O map

Table C-1 shows how the PC is typically mapped. Obviously, this list does not include every possible type of board available. You should check the boards in your system to be certain which addresses are used.

Table C-1  
PC I/O map

Address	# of Bytes	PC	XT	AT
100h to 1EFh	240	Write Only	Open	Open
1F0h to 1F8h	9	Write Only	Open	Fixed Disk
1F9h to 1FFh	7	Write Only	Open	Open
200h to 20Fh	16	Game Controller	Game Controller	Game Controller
210h to 217h	8	Open	Expansion Unit	Open
218h to 21Eh	7	Open	Open	Open
21Fh	1	Open	Reserved	Open
220h to 257h	56	Open		
258h to 25Fh	8	Intel Above Board		
260h to 277h	24	Open		
278h to 27Fh	8	LPT2		
280h to 2AFh	48	Open		
2B0h to 2DFh	48	Alternate EGA		
2E0h	1	Open		
2E1h	1	GPIB		
2E2h to 2E3h	2	Data Acquisition		
2E4h to 2F7h	20	Open		
2F8h to 2FFh	8	COM2		
300h to 31Fh	32	Prototype Area		
320h to 32Fh	16.00	Fixed Disk	Fixed Disk	Open
330h to 347h	24	Open		
348h to 357h	16	DCA 3270		
358h to 35Fh	8	Open		
360h to 36Fh	16	PC Network		
370h to 377h	8	Open		
378h to 37Fh	8	LPT1		
380h to 38Fh	16	SDLC; Binary Synchronous Communications		
390h to 393h	4	Cluster		
394h to 39Fh	12	Open		
3A0h to 3AFh	16	Binary Synchronous Communications		
3B0h to 3BFh	16	Monochrome Display Adapter		
3C0h to 3CFh	16	Enhanced Graphics Adapter		
3D0h to 3DFh	16	Color Graphics Adapter		
3E0h to 3EFh	16	Open		
3F0h to 3F7h	8	Diskette Controller		
3F8h to 3FFh	8	COM1		



Table C-2 shows how interrupts are typically mapped in a PC. Check the boards in the PC to determine exactly which interrupts are being used. The printer ports LPT1 and LPT2 have interrupts available to them, but under normal operation these interrupts are not used.

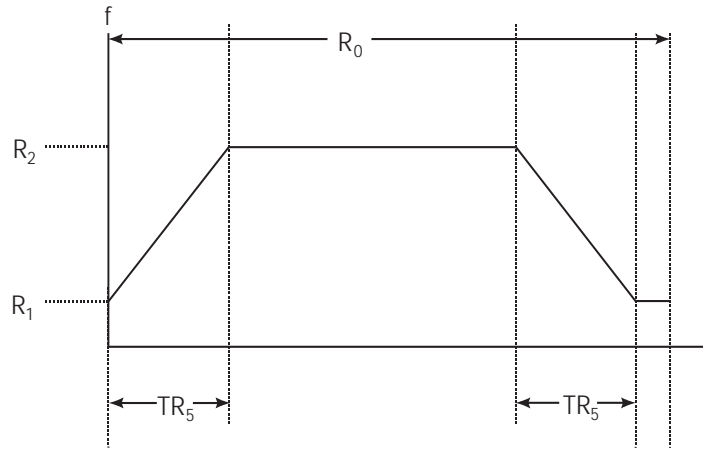
*Table C-2*  
**PC interrupt map**

<b>IRQ</b>	<b>PC</b>	<b>XT</b>	<b>AT</b>
IRQ2	Reserved	Reserved	IRQ9
	EGA Display Adapter PC Network		
IRQ3	COM2 PC Network Binary Synchronous Communications Cluster SLDC		
IRQ4	COM1 Binary Synchronous Communications SDLC		
IRQ5	Fixed Disk	Fixed Disk	LPT2
IRQ6	Floppy Disk		
IRQ7	LPT1 Cluster		
IRQ10	Not available	Not available	Open
IRQ11	Not available	Not available	Open
IRQ12	Not available	Not available	Open
IRQ14	Not available	Not available	Fixed Disk
IRQ15	Not available	Not available	Open

---

**D**  
Software  
Read/Write Technique

## Software read/write technique



Let:

$R_0 = 20,000$	(move distance in pulses)
$R_1 = 200$	(starting speed in pulses per second)
$R_2 = 2,000$	(slew speed in pulses per second)
$TR_4 = 0.2$	(acceleration time in seconds)
$TR_5 = 0.3$	(deceleration time in seconds)
$n = 1$	(1 pulse per second multiplier)
"porch"	5 pulses

Using the determined value  $n$ , you then calculate the values for  $R_7$ ,  $R_4$ ,  $R_5$ , and  $R_6$ . For example, referring to the figure above:

Given:  $f_{\text{clock}} = 5 \text{ MHz}$ .

Then:

$$R_7 = \frac{f_{\text{clock}}}{8192n} = \frac{5.0 \times 10^6}{(8192)(1)} = 610$$

$$R_4 = \frac{(f_{\text{clock}})(TR_4)}{R_2 - R_1} = \frac{(5.0 \times 10^6)(0.2)}{2000 - 200} = 556$$

$$R_5 = \frac{(f_{\text{clock}})(TR_5)}{R_2 - R_1} = \frac{(5.0 \times 10^6)(0.3)}{2000 - 200} = 833$$

$$R_6 = \frac{(R_2^2 - R_1^2)R_5}{16384(R_7)} + 5 = \frac{(4000000 - 40000)(883)}{994240} + 5 = 3323$$

Therefore:

R0 = 20,000	(total move in pulses)
R1 = 200	(base speed in pulses per second)
R2 = 2,000	(top speed in pulses per second)
R3 = <i>xxxx</i>	(not used)
R4 = 556	(acceleration in pulses per second <sup>2</sup> )
R5 = 833	(deceleration in pulses per second <sup>2</sup> )
R6 = 3,323	(ramp down preset in pulses)
R7 = 610	(multiplier)

The value of  $n$  is derived by comparing each of the desired values of R1, R2, and R3 (if applicable) with the maximum allowable value (8,192d) for these registers. If any of the desired values exceed the maximum allowed, then the multiplier becomes:

$$\frac{\text{desired value}}{\text{maximum allowed value}}$$

For example, if you want:

R1 = 300 pps
R2 = 7,936 pps
R3 = 12,287 pps

then the multiplier is calculated as:

$$n = \frac{12287}{8192} = 1.4998$$

However, it is always better to allow a little cushion and set the multiplier to a slightly higher value — in this case, say 1.75.

The adjusted values for R1, R2, and R3 then become:

$$R1 = \frac{300}{1.75} = 171$$

$$R2 = \frac{7936}{1.75} = 4535$$

$$R3 = \frac{12287}{1.75} = 7021$$

The following C code segment shows how to program the example profile using the `outp()` and `inp()` functions for axis B.

```

outp (0x300, 0x20); /*access the reset register*/
outp (0x301, 0x07); /*reset axis A, B, and C*/

outp (0x300, 0x18); /*access the sync latch register*/
outp (0x301, 0x00); /*enables all axis and clocks*/

outp (0x300, 0x08); /*access the command buffer*/
outp (0x301, 0x08); /*perform a soft reset*/

/*
B axis:

```

Note that the same routine is used over and over. Thus a preformatted subroutine should be used instead (a driver). However, this was done to example the method of entering values in the various registers using the `inp()` and `outp()` commands - not to show software expertise.

```

*/

outp (0x300, 0x08); /*access the command buffer*/
outp (0x301, 0x44); /*+ direction / preset mode*/

outp (0x300, 0x08); /*access the command buffer*/
outp (0x301, 0x80); /*point to the R0 register*/

outp (0x300, 0x09); /*get set to load LO byte*/
outp (0x301, 0x20); /*2000d = 004E20h*/

outp (0x300, 0x0A); /*get set to load MD byte*/
outp (0x301, 0x4E);

outp (0x300, 0x0B); /*get set to load HI byte*/
outp (0x301, 0x07);

outp (0x300, 0x08); /*access the command buffer*/
outp (0x301, 0x81); /*point to the R1 register*/

outp (0x300, 0x09); /*get set to load LO byte*/
outp (0x301, 0xC8); /*200d = 0000C8h*/

outp (0x300, 0x0A); /*get set to load MD byte*/
outp (0x301, 0x00);

outp (0x300, 0x0B); /*get set to load HI byte*/
outp (0x301, 0x00);

outp (0x300, 0x08); /*access the command buffer*/
outp (0x301, 0x82); /*point to the R2 register*/

outp (0x300, 0x09); /*get set to load LO byte*/
outp (0x301, 0xD0); /*2000d = 0007D0h*/

```

```
    outp (0x300, 0x0A); /*get set to load MD byte*/
    outp (0x301, 0x07);

    outp (0x300, 0x0B); /*get set to load HI byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x08); /*access the command buffer*/
    outp (0x301, 0x84); /*point to the R4 register*/

    outp (0x300, 0x09); /*get set to load LO byte*/
    outp (0x301, 0x0D); /*13d = 0000Dh*/

    outp (0x300, 0x0A); /*get set to load MD byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x0B); /*get set to load HI byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x08); /*access the command buffer*/
    outp (0x301, 0x85); /*point to the R5 register*/

    outp (0x300, 0x09); /*get set to load LO byte*/
    outp (0x301, 0xC8); /*19d = 000013h*/

    outp (0x300, 0x0A); /*get set to load MD byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x0B); /*get set to load HI byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x08); /*access the command buffer*/
    outp (0x301, 0x86); /*point to the R6 register*/

    outp (0x300, 0x09); /*get set to load LO byte*/
    outp (0x301, 0x51); /*81d = 000051h*/

    outp (0x300, 0x0A); /*get set to load MD byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x0B); /*get set to load HI byte*/
    outp (0x301, 0x00);

    outp (0x300, 0x08); /*access the command buffer*/
    outp (0x301, 0x87); /*point to the R7 register*/

    outp (0x300, 0x09); /*get set to load LO byte*/
    outp (0x301, 0x62); /*610d = 000262h*/

    outp (0x300, 0x0A); /*get set to load MD byte*/
    outp (0x301, 0x02);

    outp (0x300, 0x0B); /*get set to load HI byte*/
    outp (0x301, 0x00);
```

```
    outp (0x300, 0x08); /*access the command buffer*/
    outp (0x301, 0xC0); /*select output mode*/

    outp (0x300, 0x08); /*access command buffer*/
    outp (0x301, 0x15); /*start with FH1 SPD SEL*/

    outp (0x300, 0x08); /*access the command buffer*/

CheckMoveDone:
    if (inp(0x301) & 0x40) goto CheckOtherAxis;
    goto NextRoutine;

CheckOtherAxis:
    goto CheckMoveDone;

NextRoutine:
    /*other C code here*/
    /*
NOTE:  if it is not required to jump to specific locations then

        while (!(inp(0x300) & 0x40)) {;}
        may be used
    */
    */
```

---

**E**

Tech Bulletins and  
Application Notes



## Optimizing the scale factor, $n$

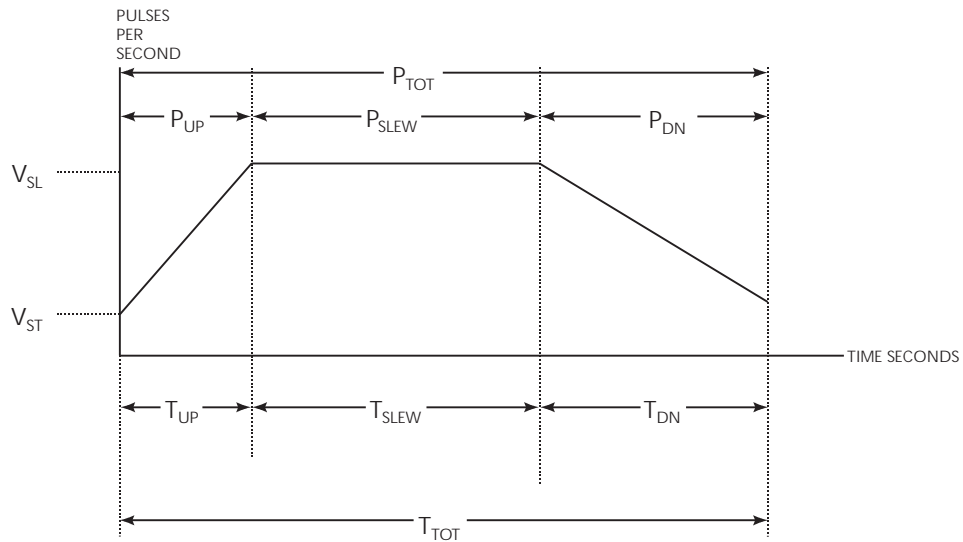
Multiplier register R7 determines the resolution of the slew velocity registers — R2 and R3. When the multiplier factor  $n$  is tuned properly, at maximum velocity, register R2 (and R3 if used) is at 1FFFh.

$$1. \quad n = \frac{\text{velocity}_{\text{slew}}}{8191}$$

Generally, stepper systems are specified with:

Total Distance	$P_{\text{tot}}$
Total Time	$T_{\text{tot}}$
Acceleration Time	$T_{\text{up}}$
Deceleration Time	$T_{\text{dn}}$
Start Velocity	$V_{\text{st}}$

From these specifications, you can obtain an optimized scale factor  $n$ .



Refer to the above figure. First, solve for total distance  $P_{\text{tot}}$  by finding the area under the curve:

$$2. \quad P_{\text{tot}} = P_{\text{up}} + P_{\text{slew}} + P_{\text{dn}}$$

$$3. \quad = \left( V_{\text{st}} T_{\text{up}} + \frac{(V_{\text{sl}} - V_{\text{st}}) T_{\text{up}}}{2} \right) + (V_{\text{sl}} (T_{\text{tot}} - T_{\text{up}} - T_{\text{dn}})) + \left( V_{\text{st}} T_{\text{dn}} + \frac{(V_{\text{sl}} - V_{\text{st}}) T_{\text{dn}}}{2} \right)$$

After simplifying and collecting like terms, you get:

$$4. \quad P_{\text{tot}} = V_{\text{sl}} (T_{\text{tot}} - \frac{1}{2} T_{\text{up}} - \frac{1}{2} T_{\text{dn}}) - \frac{V_{\text{sl}}}{2} (T_{\text{up}} + T_{\text{dn}})$$

Next, solve for  $V_{sl}$ , and simplify by factoring out the  $(1/2)$  term:

$$5. \quad V_{si} = \frac{2P_{tot} + V_{st}(T_{up} + T_{dn})}{2T_{tot} - T_{up} - T_{dn}}$$

Equation (5) is slew velocity in terms of the given specifications above.

Substitute (5) into (1):

$$6. \quad n_{opt} = \frac{2P_{tot} + V_{st}(T_{up} + T_{dn})}{8191(2T_{tot} - T_{up} - T_{dn})}$$

This is the optimum scale factor  $n$ .

To find the actual scale factor  $n_{act}$  to use, solve for R7 substituting (6) for  $n$ :

$$7. \quad R7 = \frac{f_{clock}}{8192n_{opt}}$$

Then, round down R7 to the nearest integer R7'. Back-calculate  $n_{act}$  to get the actual  $n$  to use:

$$8. \quad R7 = \frac{f_{clock}}{8192n_{opt}}$$

At this point, you can calculate the remaining register values.

The following exercise demonstrates how to calculate an optimum scale factor  $n$  with real-world values.

Given:

- $P_{tot}$  = 150,000 pulses
- $T_{tot}$  = 1.4 seconds
- $T_{up}$  = 0.2 seconds
- $T_{dn}$  = 0.2 seconds
- $V_{st}$  = 50,000 pulses per second
- $f_{clock}$  = 5 MHz.

Round down R7 to the nearest integer, 42, and back-calculate  $n_{act}$ :

$$n_{opt} = \frac{v_{si}}{8191} = \frac{2P_{tot} - V_{si}(T_{up} + T_{dn})}{8191(2T_{tot} - T_{up} - T_{dn})} = \frac{(2)(150000) - 50000(0.2 + 0.2)}{(8191)((2)(1.4) - 0.2 - 0.2)} = 14.24$$

$$R7 = \frac{f_{clock}}{8192n_{opt}} = \frac{5.0 \times 10^6}{(8192)(14.24)} = 42.85$$

$$n_{act} = \frac{f_{clock}}{8192R7} = \frac{5.0 \times 10^6}{(8192)(14.24)} = 14.532$$

This is the actual optimized  $n$  you will use.

The value to load into R1 will be:

$$R1 = \frac{V_{st}}{n_{act}} = \frac{50000}{14.532} = 3441.16 \approx 3441$$

The value to load into R2, R3 will be:

$$R2, R3 = \frac{2R0 - n_{act}R1(T_{up} + T_{dn})}{n_{act}(2T_{tot} - T_{up} - T_{dn})} = \frac{(2)(150000) - (14.532)(3441)(0.2 + 0.2)}{(14.532)((2)(1.4) - 0.2 - 0.2)} = 8028.20 \approx 8028$$

The actual slew velocity will then be:

$$V_{slew} = (R2, R3)n_{act} = (8028)(14.532) = 116662.90 \text{ pulses per second}$$

This represents the closest that you will come to the desired slew velocity.

The acceleration value in R4,R5 will then be:

$$R4, R5 = \left( \frac{f_{clock}}{(R2, R3) - R1} \right) T_{up}, T_{dn} = \left( \frac{5 \times 10^6}{8028 - 3441} \right) (0.2) = 218.01 \approx 218$$

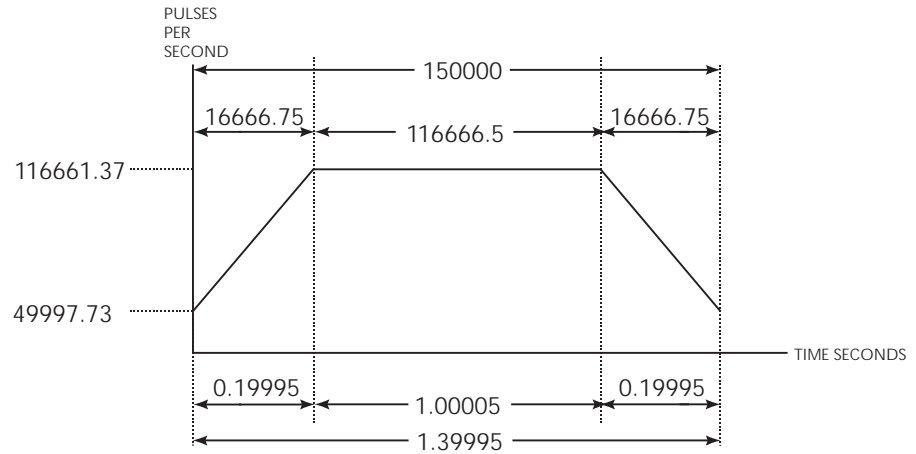
The rampdown point value in R6 will then be:

$$R6 = \left( \frac{(R2, R3)^2 R1^2}{16384} \right) \left( \frac{R5}{R7} \right) = \left( \frac{52624360}{16384} \right) \left( \frac{218}{42} \right) = 16666.39 \approx 16666$$

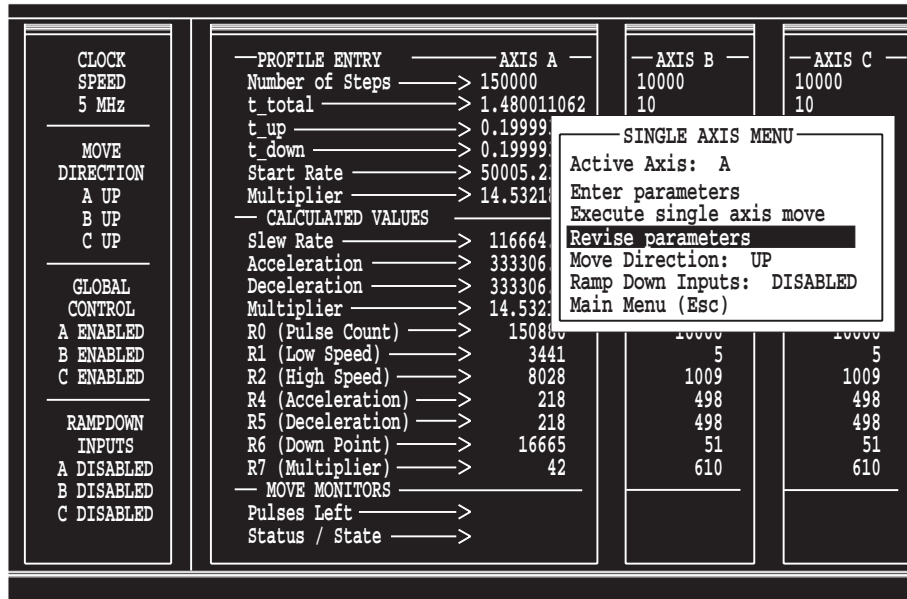
Using the same profile parameters, PRO5000 returns a *reality check* on the above numbers:

CLOCK SPEED 5 MHz  MOVE DIRECTION A UP B UP C UP  GLOBAL CONTROL A ENABLED B ENABLED C ENABLED  RAMPDOWN INPUTS A DISABLED B DISABLED C DISABLED	PROFILE ENTRY ——— AXIS A ——— Number of Steps ———> 150000 t <sub>total</sub> ———> 1.4 t <sub>up</sub> ———> .2 t <sub>down</sub> ———> .2 Start Rate ———> 50000 Multiplier ———> 14.53	——— AXIS B ——— 10000 10000	——— AXIS C ——— 10000 10000
	CALCULATED VALUES Slew Rate ———> 116664.3 Acceleration ———> 333306.8 Deceleration ———> 333306.8 Multiplier ———> 14.53218 R0 (Pulse Count) ———> 150880 R1 (Low Speed) ———> 3441 R2 (High Speed) ———> 8028 R4 (Acceleration) ———> 218 R5 (Deceleration) ———> 218 R6 (Down Point) ———> 16665 R7 (Multiplier) ———> 42	SINGLE AXIS MENU Active Axis: A Enter parameters Execute single axis move Revise Parameters Move Direction: UP RampDown Inputs: DISABLED Main Menu (Esc)	
	MOVE MONITORS Pulses Left ———> Status / State ———>	10000 5 1009 498 498 51 610	10000 5 1009 498 498 51 610

The actual profile will look like:



Using PRO5000 to back-calculate the actual physical values, we see that the actual numbers do not exactly agree with the calculated parameter values.



The differences in actual versus desired profile values are due to the floating point to integer conversion necessary for the registers. Also, stepper motors are only as precise as the detent resolution, and theoretical profiles may not correspond with this resolution. The only exceptions are microstepper and nanostepper motors whose resolution approach servo precision.

The returned values in the PRO5000 Revised Parameter screen do not exactly agree with the profile figure above due to rounding error. The optimum value for  $n$  will depend largely on the amount of precision you need. Typically, you will only need to calculate  $n$  to 3 decimal points of precision, but PRO5000 will calculate to the computing precision of your computer.

For more on PRO5000, see Appendix B of the *Model 5000 Software Developer's Guide*.

---

# **F** Revision History

## Revision /

Schematic Revision: 400088/  
Board Revision: 800048/  
Document Number:  
Description of Change: Initial release.

## Revision A

Date: 1/25/88  
Schematic Revision: 400088A  
Board Revision: 800048A  
Document Number:  
Description of Change: Bypass U3-C inverter and replace U36 with 74F32. This will improve data setup time during writes to prevent problems in slower PCs.

## Revision B

Date: 2/19/88  
Schematic Revision: 400088B  
Board Revision: 800048B  
Document Number:  
Description of Change: Corrected errors in the part list.

## Revision C

Date: 7/20/88  
Schematic Revision: 400088C  
Board Revision: 800048C  
Document Number:  
Description of Change: Replaced RP1 10Kohm resistor pack with 470 ohm pack to increase noise immunity in address decoding.

## Revision D

Date: 3/28/89  
Schematic Revision: 400088D  
Board Revision: 800048D  
Document Number:  
Description of Change: Added 0.1uF capacitor between pins 5 and 8 of U15, U17, and U19.

## Revision E

Date:	4/24/89
Schematic Revision:	400088E
Board Revision:	800048E
Document Number:	
Description of Change:	Changed resistors R3, R4, R7, R8, R11, and R12 to 2.2K, $\frac{1}{4}$ W, 5%.

## Revision F

Date:	2/20/90
Schematic Revision:	400088F
Board Revision:	800048F
Document Number:	
Description of Change:	PC board re-layed out. Removed all BUS CAP strips from PC board. Added 19-0.1uF capacitors between +5V and ground. Added 3-1.0uF capacitors C32 to C34. Other miscellaneous resistor changes.

## Revision G

Date:	11/12/90
Schematic Revision:	400088G
Board Revision:	800048G
Document Number:	
Description of Change:	Miscellaneous part number change.

## Revision H

Date:	2/19/92
Schematic Revision:	400088H
Board Revision:	800048H
Document Number:	5000TR V1.1 5000SDG V1.0
Description of Change:	Changed connector J1 from plastic Dsub to metal Dsub.



---

**G**

Introduction to  
the Model 9011  
Motion Simulator

## Features

Powered by an external or internal supply.

Simulates:

- Analog  $\pm 10$ VDC servo motor/amplifier packages with or without encoder feedback
- Pulse Width Modulated  $\pm$ PWM servo motor/amplifier packages with or without encoder feedback.
- +PWM with Direction bit servo motor/amplifier packages with or without encoder feedback.
- Stepper motor translator packages using step and direction with or without encoder feedback.
- Single ended or differential quadrature encoders with index pulse.

Operates in open or closed loop fashion.

Functions as a battery box to drive analog and PWM servo systems or stepper motor control packages.

Visual representation of encoder signals and motor rotation.

Compact size: 6×4×2 inches.

Screw terminals for easy connection.

## Application

The Model 9011 Motion Simulator is a tool designed to assist you in the development, debugging, or troubleshooting of motion control systems.

## General description

The 9011 allows you to test system software and/or motion control hardware (controller, motor/amplifier, encoder). It is designed to be a fully functional and integral part of your test and debug operations. Its usefulness is limited only by your imagination. An important point to consider is that, by using the 9011, motion control software can be tested off line. Testing software off line can prevent the prospect of machine catastrophes due to software bugs. Also, the 9011 gives you the flexibility of writing and testing your software prior to receiving the motor, amplifier, and encoder hardware. In many cases, it is not possible to test the motor hardware in the confines of your lab. Using the 9011 simulator box just once in the development, debugging, or troubleshooting of your system will generally justify your investment in this product.

## Technical specifications

Power:	External +5VDC to +15VDC (from the unit under test) Internal +6VDC (4 AA cells) 50mA current draw, maximum
Bandwidth:	488Hz
Encoder:	256 counts-per-revolution Index located at the zero count position
A/D threshold:	50mVDC
Dimensions:	6×4×2 inches
Inertia (J):	Can be controlled by use of RLC networking

## Operation

The unit is operated by a four position function switch. The ENCODER OUTPUT control (right hand control) adjusts the encoder signal frequency when the function switch is in the ENCODER position. Sixteen LEDs arranged in a circle, will light in a clockwise or counter-clockwise manner to simulate motor rotation. The quadrature encoder output is annunciated by three LEDs labeled A, B, and Z (Index), corresponding to the phase assignments of each.

A *Phoenix* screw-terminal connector is provided to allow you to connect the 9011 to your system as your application dictates. You may connect external power to the 9011 ranging from +5VDC to +15VDC, or for portability you may jumper the BAT OUT terminal to the +5V IN terminal.

You may connect the 9011 simulator to your system in a variety of ways (refer to the connection figures).

### Servo systems

With the function switch in the  $\pm$  SIGNAL position, the 9011 will accept analog or  $\pm$ PWM signals. In the PULSE/DIRECTION position, the simulator will accept a +PWM or -PWM signal with a direction bit. When using -PWM signals with a direction bit, the high inactive state voltage of the -PWM signal must be connected to REF PWR on the screw terminal.

To close the loop, the internal encoder provides differential quadrature feedback available at the screw terminal. Motor rotation and encoder pulsing will be displayed directly on the 9011. Figures G-1 through G-6 show various ways to connect the 9011 into a system.

### Stepper systems

To simulate a stepper system, place the function switch in the PULSE/DIRECTION position. Connect the stepper pulse to either the active high or active low pulse terminal and connect the direction input to DIR IN. (See Figures G-1 to G-6.) To close the loop, connect the differential quadrature feedback from the 9011 to the controller.

### Encoders

Selecting ENCODER on the function switch allows you either to simulate an encoder or to operate the 9011 as a *battery box*. Using the 9011 as a battery box enables you to drive external analog, PWM, or stepper motor systems open loop. To simulate analog, obtain the analog signal from the ANA OUT terminal. Rotating ENCODER OUTPUT will cause the analog output voltage to change accordingly. DIR O and encoder outputs — A, B, and Z — provide pulse and direction signal information for stepper and some restricted PWM operations. When testing encoders, place the function switch to the PULSE/DIRECTION position, and apply each encoder signal (one at a time) to the stepper pulse inputs, active high to pulse high and active low to pulse low.

Pulse High                      Pulse Low

When the encoder under test is rotated, the simulator LEDs will also rotate.

## Pin assignments

Table G-1 describes the connections to the 9011 through a 24-terminal Phoenix connector. The connector assignments are shown in table G-1.

Table G-2 and figures G-1 to G-6 show how to hook the 9011 in a variety of applications. In the following figures, dotted lines show alternate connections.

Table G-1

### Model 9011 connector definitions

Name	Definition
+5V IN	5VDC from external source
+15V IN	5-15VDC from external source
BAT OUT	Internal 6VDC for portable operation
PWR RET	External power supply return or COM
±SIG IN	Analog or ± PWM signal input
REF PWR	/WOM ref (active low state voltage)
/PWM IN	–PWM input signal
PWM IN	+PWM input signal
DIR IN	Direction signal input for PWM or stepper
COM†	Signal common
	Active LOW stepper pulse input signal
	Active HIGH stepper pulse input signal
COM†	Signal common
DIR O	Direction signal output (TTL)
ANA OUT	Analog output (0 to 5VDC)
A OUT	Encoder phase A output (TTL)
/A OUT	Encoder phase /A output (TTL)
COM†	Signal common
B OUT	Encoder phase B output (TTL)
/B OUT	Encoder phase /B output (TTL)
COM†	Signal common
Z OUT	Encoder phase Z output (TTL)
/ZOUT	Encoder phase /Z output (TTL)
COM†	Signal common

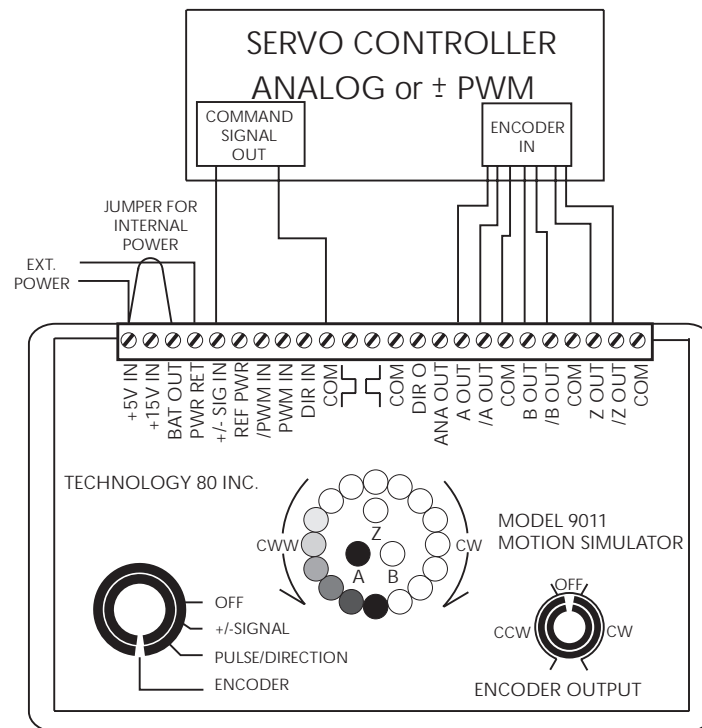
†All COM signals are electrically equivalent.

# Common applications for the Model 9011

Table G-2  
9011 Connections for common applications

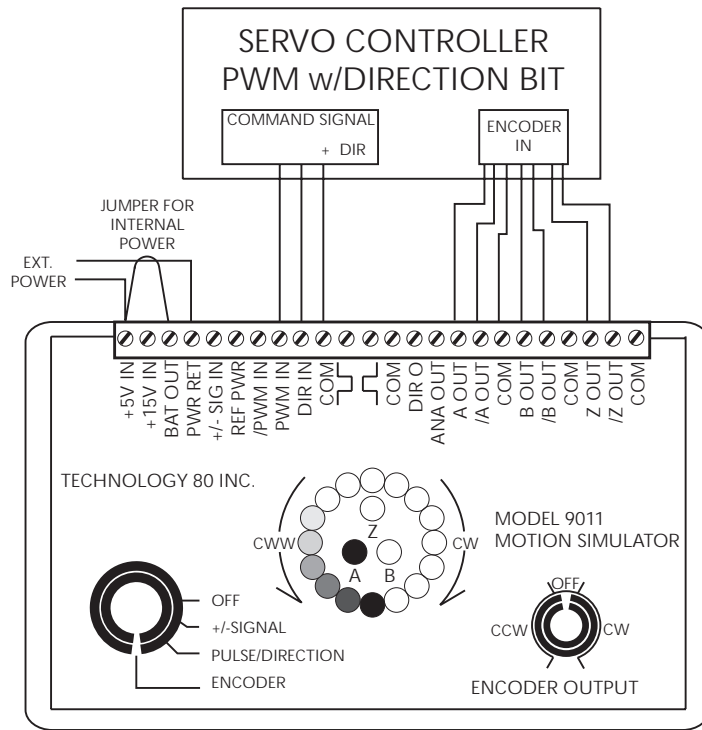
Analog		±PWM		PWM w/Sign Bit		Stepper Pulse/Direction		Encoder	
Signal	Connect	Signal	Connect	Signal	Connect	Signal	Connect	Signal	Connect
+DAC	±SIG IN	+PWM	± SIG IN	PWM	PWM IN	PULSE IN		A	A
-DAC	COM	-PWM	COM	/PWM	/PWM IN	/PULSE IN		/A	/A
				PWM	COM			B	B
				COM	DIR IN	DIRECT'N	DIR IN	/B	/B
				DIRECT'N	COM	COMMON	COM	Z	Z
				DIR COM				/Z	/Z

Figure G-1  
Analog or ±PWM application



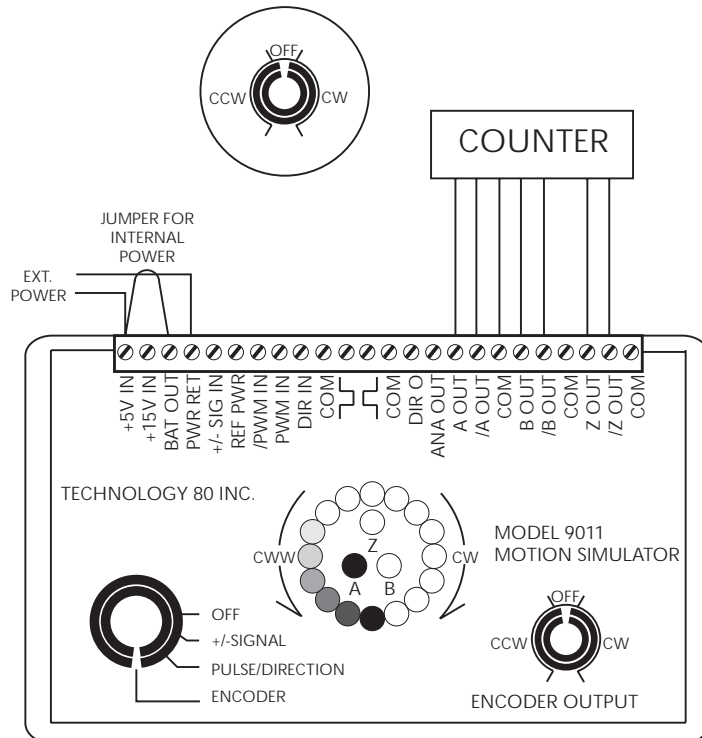
Note: This configuration simulates a closed loop system (amplifier/motor/encoder).

Figure G-2  
Pulse and direction



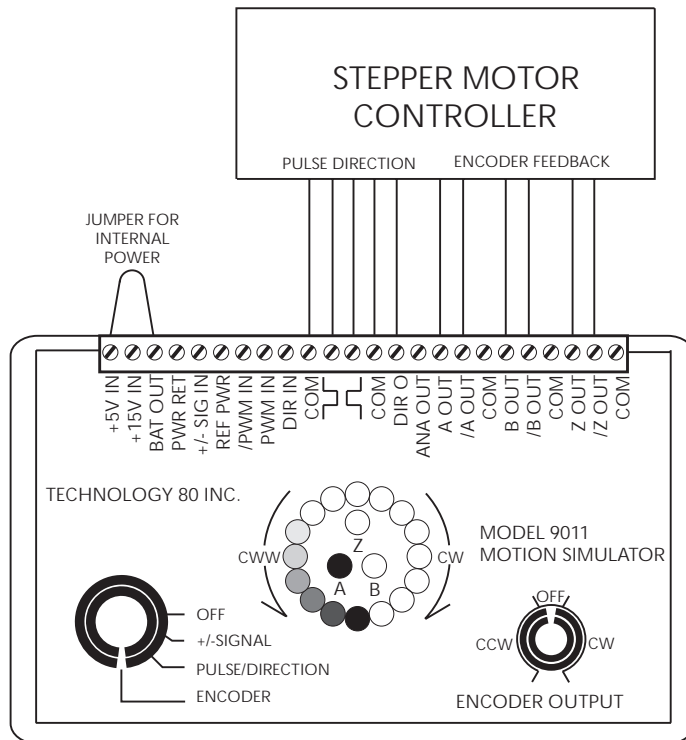
Note: You can simulate a pulse and direction closed loop system.

Figure G-3  
Using the 9011 to isolate encoder problems



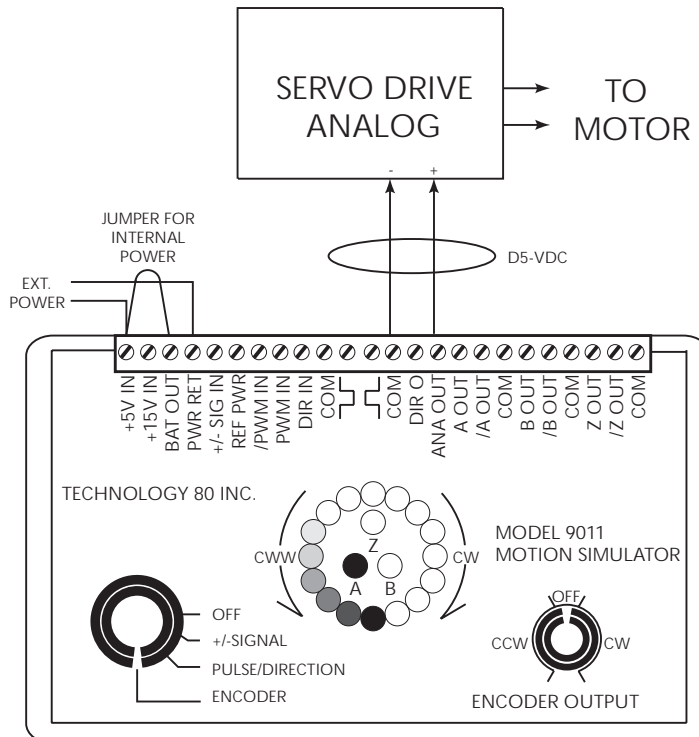
Note: You can substitute the 9011 for an encoder. The ENCODER OUTPUT control varies the rate of count.

Figure G-4  
Stepper system



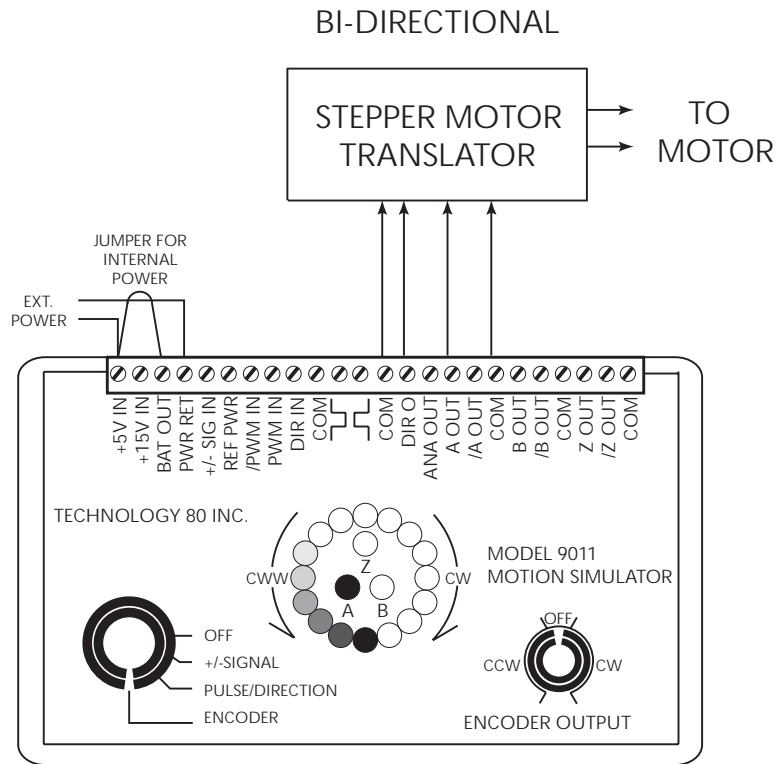
Note: This configuration shows a 9011 closing the loop in a stepper motor system.

Figure G-5  
Driving a servo amplifier



Note: The 9011 can be used (as a battery box) to provide the command to a servo system.

Figure G-6  
Driving a stepper motor system



Note: The 9011 can be used to provide the bi-directional command for a stepper motor system.



---

# H Circuit Diagrams



# Index

---

## A

- Additional operating parameters A-11
- Addressing an onboard port 2-3
- Application G-2
- Automatic rotation (equal priority) 3-5

## C

- Circuit diagrams H-1
- Command buffer 2-6
- Common applications for the Model 9011 G-5
- Completing an interrupt 3-5
- Connector pinouts 1-8
- Constant speed continuous mode A-7
- Constant speed preset mode A-5
- Constant speed return to home A-9
- Control mode selection 2-8

## D

- Data register selection 2-9
- Description of interrupt control 3-2
- Description of the 5000 1-2
- Determine R2 to R7 from given values B-3

## E

- Encoders G-3
- End-of-Interrupt command 3-4
- Example 1 B-4
- Example 2 B-5

## F

- Features G-2
- Fully nested mode 3-5

## G

- General description G-2

## H

- High speed continuous mode A-6
- High speed preset mode A-4
- High speed return to home A-8

## I

- ICW1 format and description 3-7
- ICW4 format and description 3-7
- Initialization Command Words (ICW) 3-6
- Initialization flow chart A-2
- Installation 1-3
- Interrupt control 3-1
- Interrupt Mask Register (IMR) 3-3
- Interrupt Output (INT) 3-3
- Interrupt Request Rgtr (IRR), In-Service Rgtr (ISR) 3-3
- Interrupt sequence, 80x86/80x88 mode 3-4
- Introduction and installation 1-1
- Introduction to the Model 9011 motion simulator G-1

## L

- Loading the Axis A counter 2-2

## N

- Non-vectored mode (poll command) 3-5

## O

- OCW1 format and description 3-9
- OCW2 format and description 3-9
- OCW2 commands 3-9
- OCW3 format and description 3-10
- Operating mode selection 2-6
- Operating modes 3-5
- Operation G-3
- Operation and programming 2-1
- Operation Command Words (OCW) 3-8
- Optimizing the scale factor,  $n$  E-2
- Output mode selection 2-10

## P

- PC I/O and interrupt mapping C-1
- PC I/O map C-2
- PIC operation 3-4
- PIC programming 3-6
- Pin assignments G-4
- Port locations 2-3
- Power 1-3
- Priority Resolver (PR) 3-3
- Programming 2-4

## R

- Reading from an onboard port 2-4
- Reading the state buffer 2-5
- Reading the stepper controller status port 2-5
- Register descriptions 2-11
- Reset latch 2-6
- Revision/ F-2
- Revision A F-2
- Revision B F-2
- Revision C F-2
- Revision D F-2
- Revision E F-3
- Revision F F-3
- Revision G F-3
- Revision H F-3
- Revision history F-1

## S

- Servo systems G-3
- Setting speed data A-3
- Software read/write technique D-1, D2
- Special mask mode 3-5
- Specific rotation (specific priority) 3-5
- Speed change during operation A-10
- Stepper systems G-3
- Sync latch 2-5

## T

- Tech bulletins and application notes E-1
- Technical specifications 1-2, G-2
- Theory of operation 2-2
- Typical high speed preset mode calculations B-1, B-2
- Typical operation procedures A-1

## U

- User-accessible registers 2-11

## W

- W7, board base address 1-3
- W9 to W11, clock speed select 1-5
- W8, interrupt select 1-6
- W1 to W6, opto power (bus) 1-7
- Writing the controller internal registers 2-4
- Writing to an onboard port 2-4



# Service Form

Model No. \_\_\_\_\_ Serial No. \_\_\_\_\_ Date \_\_\_\_\_

Name and Telephone No. \_\_\_\_\_

Company \_\_\_\_\_

List all control settings, describe problem and check boxes that apply to problem. \_\_\_\_\_

- |  |  |  |
|--|--|--|
| <input type="checkbox"/> Intermittent            | <input type="checkbox"/> Analog output follows display   | <input type="checkbox"/> Particular range or function bad; specify _____ |
| <input type="checkbox"/> IEEE failure            | <input type="checkbox"/> Obvious problem on power-up     | <input type="checkbox"/> Batteries and fuses are OK                      |
| <input type="checkbox"/> Front panel operational | <input type="checkbox"/> All ranges or functions are bad | <input type="checkbox"/> Checked all cables                              |

Display or output (check one)

- |                                   |  |
|-----------------------------------|--|
| <input type="checkbox"/> Drifts   | <input type="checkbox"/> Unable to zero              |
| <input type="checkbox"/> Unstable | <input type="checkbox"/> Will not read applied input |
| <input type="checkbox"/> Overload |  |

- |   |  |
|---|--|
| <input type="checkbox"/> Calibration only | <input type="checkbox"/> Certificate of calibration required |
| <input type="checkbox"/> Data required    |  |

(attach any additional sheets as necessary)

Show a block diagram of your measurement system including all instruments connected (whether power is turned on or not). Also, describe signal source.

Where is the measurement being performed? (factory, controlled laboratory, out-of-doors, etc.)

What power line voltage is used? \_\_\_\_\_ Ambient temperature? \_\_\_\_\_ °F

Relative humidity? \_\_\_\_\_ Other? \_\_\_\_\_

Any additional information. (If special modifications have been made by the user, please describe.)

Be sure to include your name and phone number on this service form.



**Keithley Instruments, Inc.**  
28775 Aurora Road  
Cleveland, Ohio 44139

Printed in the U.S.A.