



PK2500

C-Programmable Controller

User's Manual

Revision D



PK2500 User's Manual

Part Number 019-0063 • Revision D

Last revised on February 11, 2000 • Printed in U.S.A.

Copyright

© 1999 Z-World, Inc. • All rights reserved.

Z-World reserves the right to make changes and improvements to its products without providing notice.

Trademarks

- Dynamic C[®] is a registered trademark of Z-World, Inc.
 - Windows[®] is a registered trademark of Microsoft Corporation
 - PLCBus[™] is a trademark of Z-World, Inc.
-

Notice to Users

When a system failure may cause serious consequences, protecting life and property against such consequences with a backup system or safety device is essential. The buyer agrees that protection against consequences resulting from system failure is the buyer's responsibility.

This device is not approved for life-support or medical systems.

All Z-World products are 100 percent functionally tested. Additional testing may include visual quality control inspections or mechanical defects analyzer inspections. Specifications are based on characterization of tested sample units rather than testing over temperature and voltage of each unit. Z-World may qualify components to operate within a range of parameters that is different from the manufacturer's recommended range. This strategy is believed to be more economical and effective. Additional testing or burn-in of an individual unit is available by special arrangement.

Company Address

Z-World, Inc.

2900 Spafford Street
Davis, California 95616-6800
USA



Telephone: (530) 757-3737
Facsimile: (530) 753-5141
Web Site: <http://www.zworld.com>
E-Mail: zworld@zworld.com

TABLE OF CONTENTS

About This Manual	vii
Chapter 1: Overview	11
PK2500 Overview	12
Features	13
Core Module	13
Other Features	13
Optional Accessories	14
Software Development and Evaluation Tools	14
CE Compliance	15
Chapter 2: Getting Started	17
Initial PK2500 Setup	18
Parts Required	18
Connecting the PK2500 to a Host PC	18
Running Dynamic C	20
Test the Communication Line	20
Selecting Communications Rate, Port, and Protocol	22
Running a Sample Program	22
Chapter 3: Input/Output Configuration	23
PK2500 Inputs and Outputs	24
Flexible Inputs/Outputs	24
Configuring Serial Communications	27
Configuring Inputs and Outputs	28
Protected Digital Inputs vs. High-Current Outputs	28
RS-485 vs. Protected Digital Inputs	29
A/D Converter Input vs. A/D Voltage Reference Output	29
PK2500 Subsystems	30
Chapter 4: System Development	31
PK2500 Operating Modes	32
Changing the PK2500's Operating Mode	33
Running a Program	34

Using Digital Inputs/Outputs	35
Protected Digital Inputs	35
How to Read the Inputs	35
High-Current Outputs	37
How to Use the Outputs	37
Serial Communication	40
RS-232 Communication	40
RS-232 Connector Pinouts	40
RS-485 Network	41
Termination and Bias Resistors	42
Relay Outputs	43
How to Use the Relay Outputs	44
Analog-to-Digital Converter Inputs	46
Scaling Input Range	48
Theory of Operation	50
The VOFF Voltage Divider	51
DC Gain	51
Finding VOFF	52
Practical Considerations	53
Input Impedance	54
Frequency Response	54
How to Use the Analog-to-Digital Converter	55
Using the Analog Inputs	55
Using the A/D Voltage Reference	56
Additional Features	57
Pulse-Width Modulated (PWM) Outputs	57
How to Use the PWM Feature	57
User-Programmable LEDs	60
Real-Time Clock (RTC)	61
Power Supervisor	61
+5 V Output	62

Chapter 5: **Software Reference** **63**

Input/Output Software Drivers	64
Digital Inputs/Outputs	64
Level-Sensitive Interrupts	66
Interrupt Service Routines	67
Pulse-Width Modulation Outputs	68
References to Additional Software Features	69

Advanced Input/Output Programming	70
Digital Input Addressing Details	71
Digital Output and Relay Output Addressing Details	72
Analog-to-Digital Converter Addressing Details	73
LED Addressing Details	74
RS-485 Driver Addressing Details	74
PWM Addressing Details	74
PWM Advanced Programming Functions	78
Appendix A: Troubleshooting	81
Out of the Box	82
Dynamic C Will Not Start	83
Dynamic C Loses Serial Link	83
PK2500 Repeatedly Resets	83
Common Programming Errors	84
Appendix B: Specifications	85
Electrical and Mechanical Specifications	86
Factory Default Jumper Positions	88
Protected Digital Inputs	90
Frequency Response for IN-00 to IN-05, and IN-08 to IN-15	90
Frequency Response for IN-06 and IN-07	91
Customization	91
Frequency Response and Input Range	91
Default Pull-Up Assignments	92
High-Current Drivers	92
“KA” and “KB”	93
Appendix C: Power Management	95
Power Failure Sequence of Events	96
Power-Failure Detection Circuitry	96
Appendix D: Sinking vs. Sourcing Drivers	99
Selecting Sourcing or Sinking Drivers	101
Sinking Driver (Low-Side Drive)	101
Sourcing Driver (High-Side Drive)	102
Appendix E: Serial Interface Board 2	103
Introduction	104
External Dimensions	105

Appendix F: **Enclosure Mounting** **107**

Appendix G: **Nonvolatile Storage** **111**

Appendix H: **I/O Map and Interrupt Vectors** **113**

- PK2500 Input/Output Map 114
- Real-Time Clock Registers 116
- Other Input/Output Addresses 117
- Interrupt Vectors 118
- Interrupt Priorities 119

Appendix I: **Battery** **121**

- Storage Conditions and Shelf Life 122
- Instructions for Replacing the Lithium Battery 122
- Battery Cautions 123

Index **125**

ABOUT THIS MANUAL

This manual provides instructions for installing, testing, configuring, and interconnecting the Z-World PK2500 controller. Instructions are also provided for using Dynamic C[®] functions.

Assumptions

Assumptions are made regarding the user's knowledge and experience in the following areas.

- Ability to design and engineer the target system that a PK2500 will control.
- Understanding of the basics of operating a software program and editing files under Windows on a PC.
- Knowledge of the basics of C programming.



For a full treatment of C, refer to the following texts.

The C Programming Language by Kernighan and Ritchie
and/or

C: A Reference Manual by Harbison and Steel

- Knowledge of basic Z80 assembly language and architecture.



For documentation from Zilog, refer to the following texts.

Z180 MPU User's Manual

Z180 Serial Communication Controllers

Z80 Microprocessor Family User's Manual

Acronyms

Table 1 lists and defines the acronyms that may be used in this manual.







Table 1. Acronyms

Acronym	Meaning
EPROM	Erasable Programmable Read-Only Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
LCD	Liquid Crystal Display
LED	Light-Emitting Diode
NMI	Nonmaskable Interrupt
PIO	Parallel Input/Output Circuit (Individually Programmable Input/Output)
PRT	Programmable Reload Timer
RAM	Random Access Memory
RTC	Real-Time Clock
SIB	Serial Interface Board
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver Transmitter

Icons

Table 2 displays and defines icons that may be used in this manual.

Table 2. Icons

Icon	Meaning	Icon	Meaning
	Refer to or see		Note
	Please contact	Tip	Tip
	Caution		High Voltage
	Factory Default		

Conventions

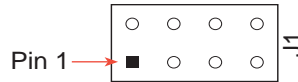
Table 3 lists and defines the typographic conventions that may be used in this manual.

Table 3. Typographic Conventions

Example	Description
while	Courier font (bold) indicates a program, a fragment of a program, or a Dynamic C keyword or phrase.
// IN-01...	Program comments are written in Courier font, plain face.
<i>Italics</i>	Indicates that something should be typed instead of the italicized words (e.g., in place of <i>filename</i> , type a file's name).
Edit	Sans serif font (bold) signifies a menu or menu selection.
...	An ellipsis indicates that (1) irrelevant program text is omitted for brevity or that (2) preceding program text may be repeated indefinitely.
[]	Brackets in a C function's definition or program segment indicate that the enclosed directive is optional.
< >	Angle brackets occasionally enclose classes of terms.
a b c	A vertical bar indicates that a choice should be made from among the items listed.

Pin Number 1

A black square indicates pin 1 of all headers.



Measurements

All diagram and graphic measurements are in inches followed by millimeters enclosed in parenthesis.

Blank



CHAPTER 1: **OVERVIEW**

Chapter 1 provides a comprehensive overview and description of the PK2500. The following sections are included.

- PK2500 Overview
- Features
- Flexibility and Customization
- Optional Accessories
- Software Development and Evaluation Tools
- CE Compliance

PK2500 Overview

The PK2500 is a small, powerful, and extremely flexible system controller. The PK2500 consists of a main board with a core module to provide the processing power. The core module provides the processor, real time clock, supervisor, memory, and control of the various inputs/outputs.

A rugged enclosure houses the PK2500. The base plate is compatible with multiple mounting options, including DIN rail.

The PK2500 is programmed using Dynamic C, Z-World's version of the C programming language.

Figure 1-1 illustrates the PK2500 component layout.

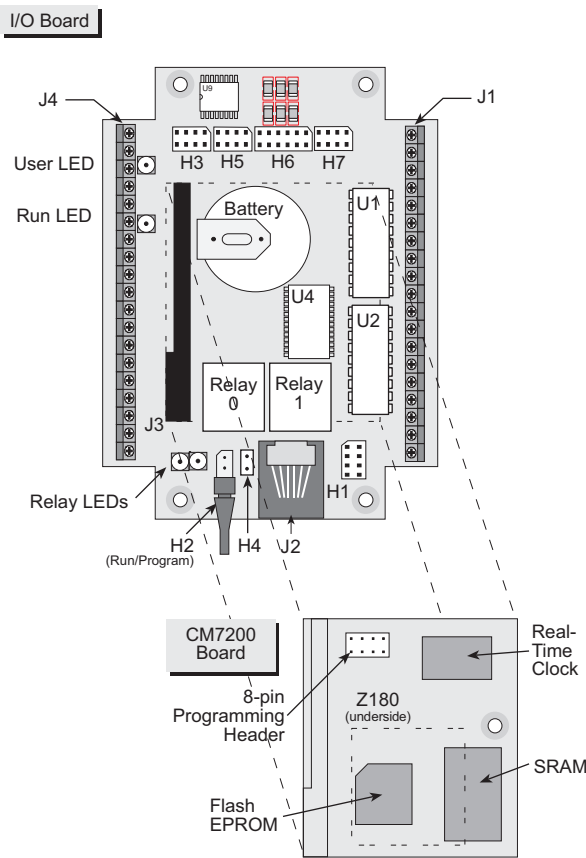


Figure 1-1. PK2500 Component Layout (Top View, Case Removed)

Features

The PK2500 offers the following features.

Core Module

- Z180 microprocessor operating at 18.432 MHz, 128K SRAM, and 128K flash EPROM.



The core module used on the PK2500 is a CM7200 that has been modified to include a header at location H1 on the CM7200. The BIOS is also different from that supplied with regular stock CM7200s.

Other Features

- Thirty pins (out of the 38 total) are available for assigning various I/O combinations. One RJ-12 modular jack is also provided for RS-232 communication.
- Factory-configured with 10 protected digital inputs, two of which have software-assignable level-sensitive interrupts, and two of which may be configured as an RS-485 serial port. Up to six additional protected digital inputs are available by changing the configuration of the high-current outputs.
- Factory-configured with 12 high-current driver outputs in two banks of six each. The outputs normally have sinking drivers, but sinking and optional sourcing drivers may be mixed between the two banks for push-pull or H-bridge operation. Three of the outputs on each of the two banks may be configured as protected digital inputs.
- Four 12-bit analog/digital converter inputs.
- Two SPST relay outputs.
- Two RS-232 serial communication ports are available. One RS-485 serial port is available by reconfiguring two protected digital inputs.
- One bank of six of the digital outputs can optionally provide pulse-width modulation under software control.
- Switching +5 V voltage regulator with some excess capacity for external loads.
- Two programmable LEDs.
- Quick-disconnect screw terminal blocks with a 3.5 mm pitch.

Table 1-1 lists the versions of the PK2500 that are available.


Table 1-1. PK2500 Series Features

Model	Features
PK2500	Standard full-featured model
PK2510	PK2500 with 9.216 MHz clock, 32K SRAM, and fixed connectors

The PK2500 is also available without the enclosure.

The PK2500 was designed with flexibility in mind. Nine of the input/output assignments are assignable via jumpers, allowing quick tailoring to specific I/O configurations.

The circuit layout of the PK2500 has been optimized for quick-turn custom manufacturing. Automated surface-mount manufacturing can deliver a PK2500 controller with the *exact* hardware configuration required by the application.

 Appendix B provides detailed specifications for the PK2500.

Optional Accessories

The following accessories are available for the PK2500.

- Development Kit containing manual with schematics, programming cable, AC adapter, and sourcing high-current driver chip.
- DIN Rail mounting kit.
- Sourcing driver kit.
- Serial Interface Board 2 to allow programming through a special programming port, leaving the serial channels available for the application.



For ordering information, or for more details about the various options and prices, call your Z-World Sales Representative at (530) 757-3737.

Software Development and Evaluation Tools

Dynamic C, Z-World's Windows-based real-time C language development system, is used to develop software for the PK2500. The host PC downloads the executable code through the Serial Interface Board 2 or one of the serial ports to the flash EPROM.



Z-World's Dynamic C reference manuals provide complete software descriptions and programming instructions.

CE Compliance

The PK2500 has been tested by an approved competent body, and was found to be in conformity with applicable EN and equivalent standards. Note the following requirements for incorporating the PK2500 in your application to comply with CE requirements.



- The power supply provided with the Development Kit is for development purposes only. It is the customer's responsibility to provide a clean DC supply to the controller for all applications in end-products.
- The PK2500 has been tested to Light Industrial Immunity standards. Additional shielding or filtering may be required for an industrial environment.
- The PK2500 has been tested to EN55022 Class A emission standards. Additional shielding or filtering may be required to meet Class B emission standards.



Visit the “Technical Reference” pages of the Z-World Web site at <http://www.zworld.com> for more information on shielding and filtering.

Blank



CHAPTER 2: **GETTING STARTED**

Chapter 2 provides instructions for connecting the PK2500 to a host PC, and running a sample program. The following sections are included.

- Initial PK2500 Setup
- Connecting the PK2500 to a Host PC
- Running Dynamic C
- Running a Sample Program

Initial PK2500 Setup

Parts Required

- 12 V, 500 mA unregulated DC power supply
- Programming cable

Connecting the PK2500 to a Host PC

The PK2500 may be programmed directly through its serial port or through a Serial Interface Board 2 (SIB2).

1. Connect the power supply to the +DC and GND terminals as shown in Figure 2-1.

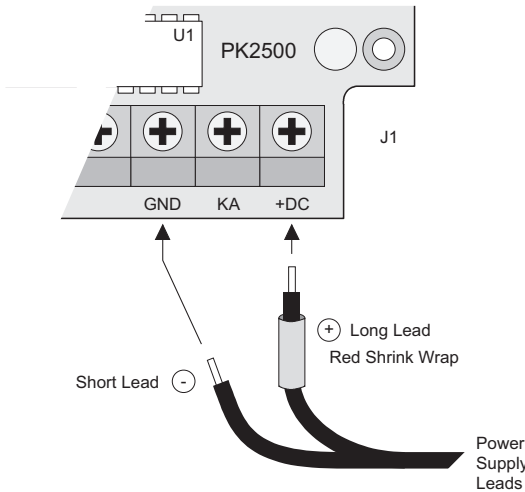


Figure 2-1. Power Supply Connections



Do not plug the transformer into the wall until all the connections and jumpers have been set.



If a transformer other than the one supplied with the Developer's Kit is used, ensure that the input voltage specifications (9 V to 40 V DC) are not exceeded. Appendix B contains complete specifications for the PK2500.

2. Ensure that the Run/Program jumper (H2) is installed as shown in Figure 2-2.

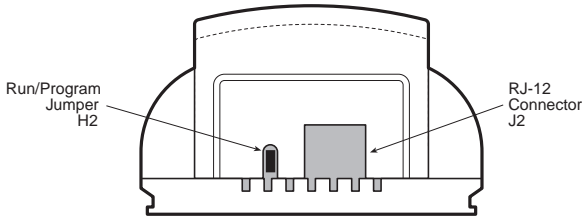


Figure 2-2. H2 Run/Program Jumper Location



Pay particular attention to the installation of the Run/Program jumper at H2. It is possible to install the jumper so that the pins are not connected. The PK2500 will not change modes if this jumper is installed incorrectly.

3. Establish a serial communication link

Option 1—Via RS-232 Serial Port

Use the adapter and the programming cable supplied with the developer's kit to connect the PK2500's RJ-12 (J2) socket to the appropriate computer COM port as shown in Figure 2-3.

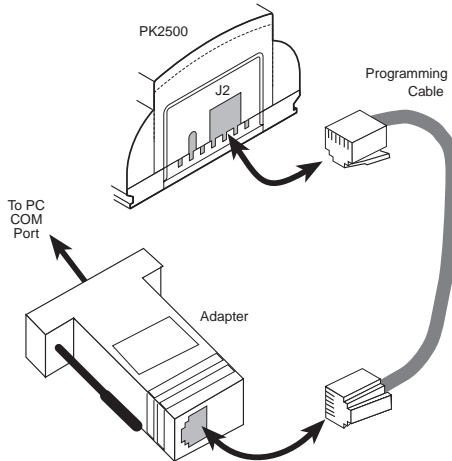


Figure 2-3. Programming Connections to RS-232 Port



Use the programming cable supplied with the Developer's Kit to avoid problems with mismatched plugs.

Option 2—Via Optional Serial Interface Board 2 (SIB2)

The SIB2 is an optional development tool that allows both serial ports to be available to an application. The PK2500 uses a Z-World CM7200 core module as the microprocessor core. The CM7200 has an additional programming port, JP1, that connects to the SIB via a 2 mm, 8-pin connector, bypassing the PK2500's RJ-12 modular jack (J2).

Connect an RJ-12 cable between the SIB2 and the RJ-12/DB9 adapter attached to the host PC (see Figure 2-4).

Plug the SIB2's 8-pin connector into header JP1 located on the CM7200.



Observe the polarity of the cable used to connect the SIB2 to JP1. The side of the cable closest to Pin 1 is marked in blue, as indicated in Figure 2-4.

4. The PK2500 is now ready for programming. The transformer may be plugged in.

Running Dynamic C

Test the Communication Line

Double-click the Dynamic C icon to start the software. Note that the PC attempts to communicate with the PK2500 each time Dynamic C is started. No error messages are displayed once communication is established.



See Appendix A, “Troubleshooting,” if an error message such as **Target Not Responding** or **Communication Error** appears.



Once the necessary changes have been made to establish communication between the host PC and the PK2500, use the Dynamic C shortcut **<Ctrl Y>** to reset the controller and initiate communication.

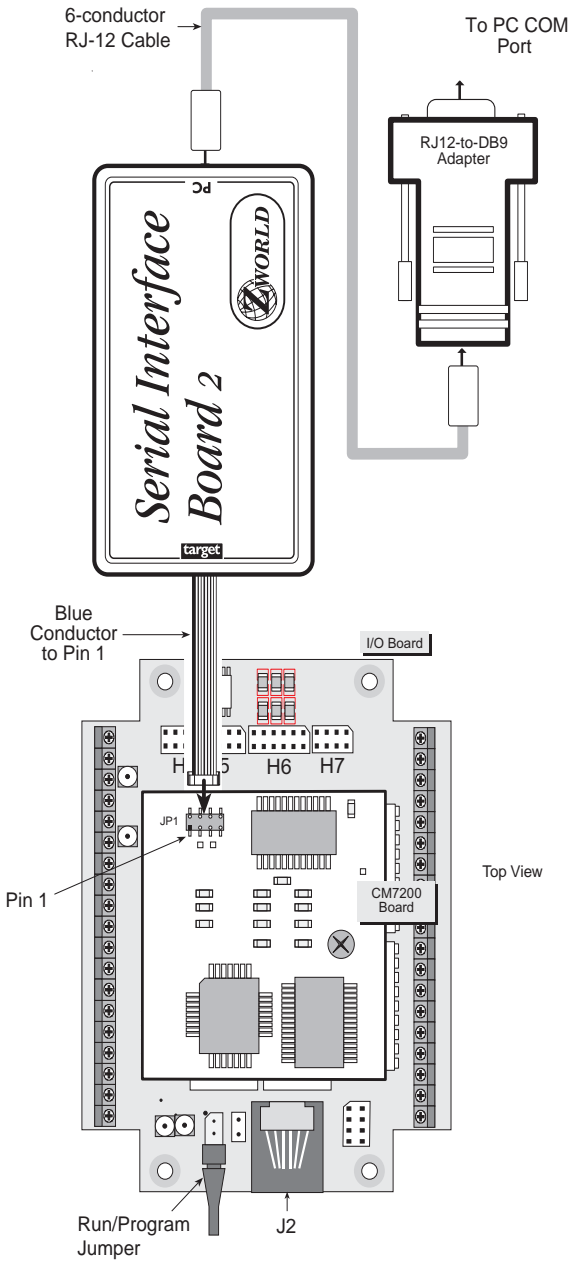


Figure 2-4. Programming Connections to Serial Interface Board 2

Selecting Communications Rate, Port, and Protocol

The communication rate, port, and protocol are all selected by choosing **Serial Options** from the Dynamic C **OPTIONS** menu.

The PK2500 supports a communication baud rate up to 57,600 bps. However, the Dynamic C software shipped by Z-World may be initialized for a different baud rate. To begin, select a communication rate of 57,600 bps. A slower rate may be required, in which case the PK2500 should be reset with a **<Ctrl Y>** after the lower rate is selected. The SIB2 automatically adjusts to the PC's communication baud rate to 9600 bps, 19,200 bps, 28,800 bps, or 57,600 bps.

Make sure that the PC serial port used to connect the serial cable (COM1 or COM2) is the one selected in the Dynamic C **OPTIONS** menu. Select the 1-stop-bit protocol.

Running a Sample Program

The sample program **FLASHLED.C** is supplied in the Dynamic C **SAMPLES\PK25xx** subdirectory. This program flashes the LED on the board.

Prior to running this test, be sure to set the communications parameters in Dynamic C so that the PC and the PK2500 are handshaking properly.

1. Compile the program by pressing **F3** or by choosing **Compile** from the **COMPILE** menu. Dynamic C compiles and downloads the program.
2. Run the program by pressing **F9** or by choosing **Run** from the **RUN** menu. The LEDs on the PK2500 will begin flashing at differing rates.
4. Press **<Ctrl Z>** to stop execution of the program.
5. If needed, press **F9** to restart execution of the program.

The Dynamic C **SAMPLES\PK25xx** subdirectory contains other sample programs that illustrate the features of the PK2500. These programs may be used as a basis for new applications.



CHAPTER 3: **INPUT/OUTPUT CONFIGURATION**

Chapter 3 describes the built-in flexibility of the PK2500 controller and describes how to configure the available inputs/outputs. The following sections are included.

- PK2500 Inputs and Outputs
- Configuring Serial Communications
- Configuring Inputs and Outputs
- PK2500 Subsystems

PK2500 Inputs and Outputs

The PK2500 provides six types of inputs/outputs (I/O).

- Protected digital inputs
- High-current driver outputs
- Analog to digital converter inputs
- Serial communication channels
- Relay outputs
- LEDs

Flexible Inputs/Outputs

The PK2500 has 30 pins dedicated to I/O. Some pins have fixed functions, others can be configured using jumpers. The inputs and outputs may be mixed and matched to suit a particular application. Figure 3-1 shows the signal names for the pins and Table 3-1 and Table 3-2 list all possible I/O function combinations.

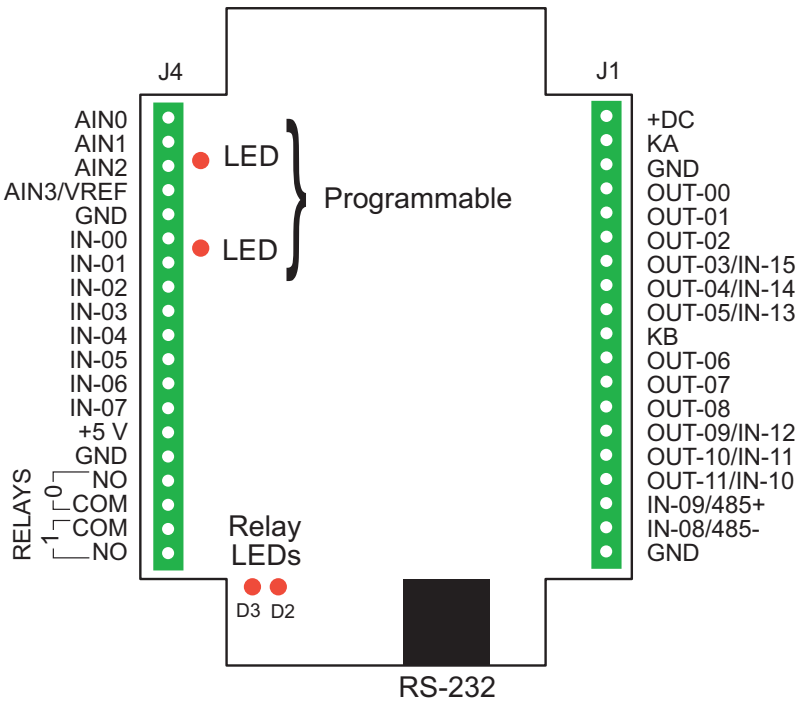




Figure 3-1. PK2500 Pinout

Table 3-1. PK2500 Header J4 I/O Functions

Pin	Label	Function 1 	Function 2 (user assignable)
1	AIN0	A/D Converter Input 0	n/a
2	AIN1	A/D Converter Input 1	n/a
3	AIN2	A/D Converter Input 2	n/a
4	AIN3/VREF	A/D Converter Input 3	A/D Voltage Reference Output
5	GND	Analog Ground	n/a
6	IN-00	Protected Digital Input 0	n/a
7	IN-01	Protected Digital Input 1	n/a
8	IN-02	Protected Digital Input 2	n/a
9	IN-03	Protected Digital Input 3	n/a
10	IN-04	Protected Digital Input 4	n/a
11	IN-05	Protected Digital Input 5	n/a
12	IN-06	Protected Digital Input 6	Protected Digital Input 6 with level-sensitive interrupt (/INT0)*
13	IN-07	Protected Digital Input 7	Protected Digital Input 7 with level-sensitive interrupt (/INT1)*
14	+5 V	+5 V	n/a
15	GND	Ground	n/a
16	RELAYS-0-NO	Relay 0, normally open	n/a
17	RELAYS-0-COM	Relay 0, common	n/a
18	RELAYS-1-COM	Relay 1, common	n/a
19	RELAYS-1-NO	Relay 2, normally open	n/a

* These interrupts are software-assignable. See Chapter 5, "Software Reference," for further details.

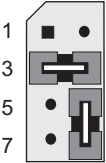
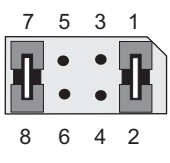

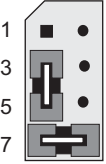
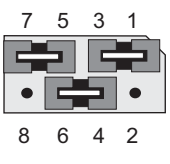
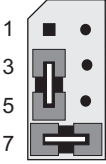
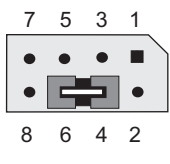
Table 3-2. PK2500 Header J1 I/O Functions

Pin	Label	Function 1	Function 2 (user assignable)
			
1	+DC	Power Supply Input	n/a
2	KA	High-Current Driver 0-5 Supply	n/a
3	GND	Ground	n/a
4	OUT-00	High-Current Output 0	n/a
5	OUT-01	High-Current Output 1	n/a
6	OUT-02	High-Current Output 2	n/a
7	OUT-03/IN-15	High-Current Output 3	Protected Digital Input 15
8	OUT-04/IN-14	High-Current Output 4	Protected Digital Input 14
9	OUT-05/IN-13	High-Current Output 5	Protected Digital Input 13
10	KB	High Current Driver 6-11 supply	n/a
11	OUT-06	High-Current Output 6	n/a
12	OUT-07	High-Current Output 7	n/a
13	OUT-08	High-Current Output 8	n/a
14	OUT-09/IN-12	High-Current Output 9	Protected Digital Input 12
15	OUT-10/IN-11	High-Current Output 10	Protected Digital Input 11
16	OUT-11/IN-10	High-Current Output 11	Protected Digital Input 10
17	IN-09/485+	Protected Digital Input 09	+RS-485 Serial Communication
18	IN-08/485-	Protected Digital Input 08	-RS-485 Serial Communication
19	GND	Ground	n/a

Configuring Serial Communications

The PK2500 has two serial-communication ports. Table 3-3 provides the three *mutually exclusive* combinations of the RS-232 and RS-485 serial protocols that can be applied to the ports.

Table 3-3. PK2500 Serial Communication Configurations

Header Jumpers	Configurations
<p style="text-align: center;">Configuration I</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>H1</p>  </div> <div style="text-align: center;"> <p>H3</p>  </div> </div>	<p>Two 3-wire RS-232 (no handshaking)</p> <div style="text-align: center;">  </div>
<p style="text-align: center;">Configuration II</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>H1</p>  </div> <div style="text-align: center;"> <p>H3</p>  </div> </div>	<p>One 5-wire RS-232 (RTS/CTS handshaking) and one RS-485</p>
<p style="text-align: center;">Configuration III</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>H1</p>  </div> <div style="text-align: center;"> <p>H3</p>  </div> </div>	<p>One 3-wire RS-232 (no handshaking) and one RS-485</p>

Configuring Inputs and Outputs

This section describes how to configure the I/O to specific application requirements.

Protected Digital Inputs vs. High-Current Outputs

This section provides information for setting the jumpers on header H6 to configure pins 7–9 and 14–16 on header J1 as high-current outputs (OUT-03 to OUT-05 and OUT-09 to OUT-11) or as protected digital inputs (IN-10 to IN-15).

Figure 3-2 illustrates the jumper settings for header H6.

- When a jumper is installed, the corresponding pin is configured as a protected digital input.
- When a jumper is *not* installed, the corresponding pin is configured as a high-current output.

Each channel may be set up individually as desired.

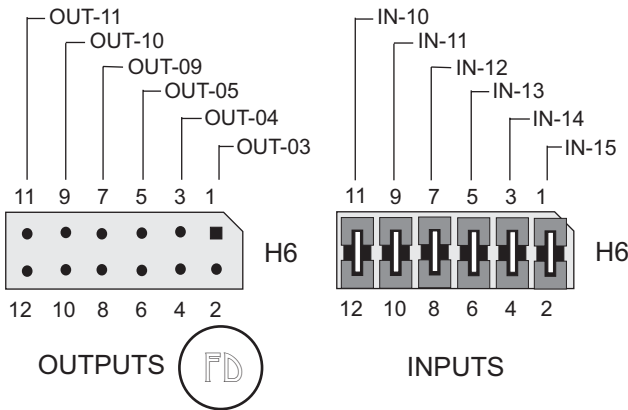


Figure 3-2. Setting PK2500 Header H6 for Protected Digital Inputs or High-Current Outputs



The high-current driver output remains hardwired to the header J1 pins when a jumper is installed on header H6 to configure these pins as protected digital inputs. Do not enable the output drivers on any of the channels configured as protected digital inputs via header H6 unless the appropriate jumpers have been removed to configure these pins as high-current outputs.

RS-485 vs. Protected Digital Inputs

This section provides information for setting the jumpers on header H1 to configure pins 17 and 18 on header J1 as protected digital inputs (IN-08 and IN-09) or as an RS-485 communications port (RS-485+ and RS-485-).

Figure 3-3 illustrates the jumper settings for header H1.

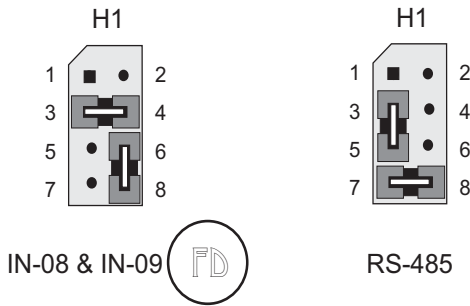


Figure 3-3. Setting PK2500 Header H1 for Protected Digital Inputs or RS-485

A/D Converter Input vs. A/D Voltage Reference Output

This section provides information for setting the jumpers on header H5 to configure pin 4 on header J4 as an A/D converter input (AIN3) or as an A/D voltage reference output (VREF).

Figure 3-4 illustrates the jumper settings for header H5.

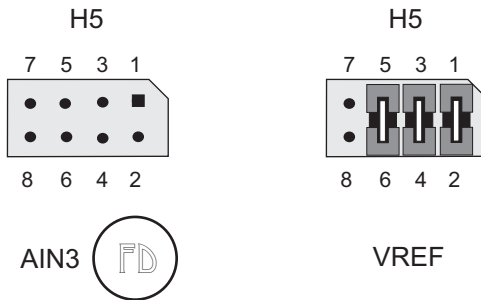


Figure 3-4. Setting PK2500 Header H5 for A/D Converter Input or A/D Voltage Reference Output

PK2500 Subsystems

Figure 3-5 summarizes the PK2500's subsystems.

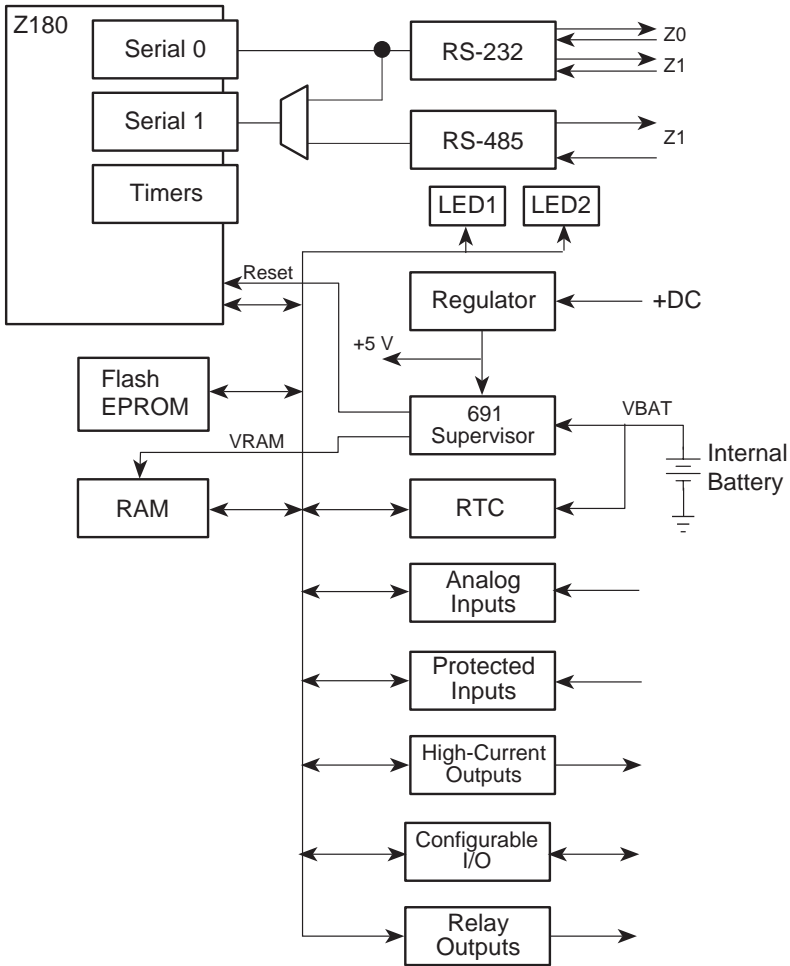


Figure 3-5. PK2500 Subsystems



CHAPTER 4: **SYSTEM DEVELOPMENT**

Chapter 4 describes how to use the features of the PK2500 controller. The following major sections are included.

- PK2500 Operating Modes
- Running a Program
- Using Digital Inputs/Outputs
 - Protected Digital Inputs
 - High-Current Outputs
 - Serial Communication
- Relay Outputs
- Analog-to-Digital Converter Inputs
- Additional Features
 - Pulse-Width Modulated Outputs
 - User-Programmable LEDs
 - Real-Time Clock (RTC)
 - Power Supervisor
 - +5 V Output

PK2500 Operating Modes

The PK2500 has two *mutually exclusive* operating modes, only one of which responds to Dynamic C. Each mode is explained in detail below.

- **Program Mode**

In Program Mode, the PK2500 runs under the control of the PC, which is running Dynamic C. The PK2500 must be in this mode to compile a program or to debug a program.

In Program Mode, the PK2500 matches the baud rate of the PC COM port up to 57,600 bps. Baud rates of 9600 bps, 19,200 bps, 28,800 bps, and 57,600 bps are possible. The User LED, shown in Figure 4-1, is enabled to remain on continuously; the Run LED is disabled, or off. Both LEDs are available for the application being developed.

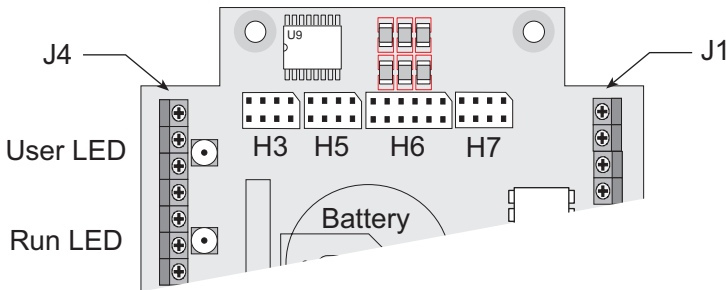


Figure 4-1. Run and User LED Locations

- **Run Mode**

In Run Mode, the PK2500 controller runs *standalone*. At power-up, the PK2500 checks to see if its onboard memory contains a program. If such a program exists, the PK2500 executes the program immediately after power-up.

Both LEDs are under the control of the application while the PK2500 is in Run Mode. The default is for the Run LED to be off and for the User LED to be on.



The PK2500 in Run Mode will not respond to Dynamic C running on a PC. Programs cannot be compiled or debugged while the PK2500 is in Run Mode.



In Run Mode, the PK2500 takes approximately 60 ms to boot up and begin execution of a program in the flash EPROM.

Table 4-1 lists the PK2500 activities in the two modes.

Table 4-1. PK2500 Activities While in Program Mode or in Run Mode

Operating Mode	Jumper H2	Activities
Program Mode	Installed	<ul style="list-style-type: none">• Compile a program• Run a program under debugger control• Run a program without “polling.” See Dynamic C manuals for a description of program polling.
Run Mode	Removed	Run program (application)

Changing the PK2500’s Operating Mode

These steps describe how to change the PK2500’s operating mode.

1. Disconnect power from the PK2500.
2. Locate the Run/Program jumper at H2 that protrudes from the board next to the RJ-12 connector (J2). Figure 4-2 shows the location of jumper H2.

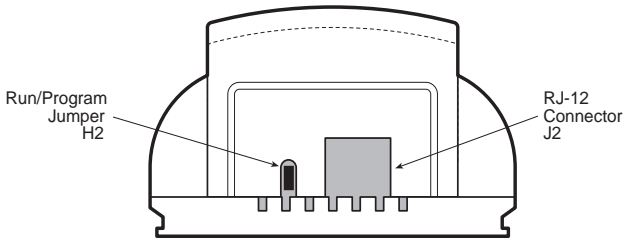


Figure 4-2. Location of Run/Program Jumper at H2

3. Select Run Mode or Program Mode.
 - Install the jumper at H2 to select Program Mode.
 - Remove jumper at H2 to select Run Mode.
4. Reapply power to restart the PK2500 in the selected mode.



Pay particular attention to the installation of the Run/Program jumper at H2. It is possible to install the jumper so that the pins are not connected. The PK2500 will not change modes if the Run/Program jumper is installed incorrectly.

Running a Program

1. Place the PK2500 in Program Mode (with the Run/Program jumper installed at H2) and cycle the unit's power.
2. Open a program if one is not already open.
3. Select the **Compile** command from the **Compile** menu, or press **F3**.
4. If no errors are detected, Dynamic C compiles the program and automatically downloads it into the PK2500's onboard flash EPROM memory.
5. Remove power from the PK2500.
6. Remove the Run/Program jumper.
7. Reapply power to the PK2500. This resets the PK2500 in the Run Mode, and the downloaded program begins to run.

The program is now loaded permanently in the PK2500's onboard flash EPROM. This program will now run automatically every time the PK2500 powers up in Run Mode until another program is loaded.



The flash EPROM has a rated lifetime of only 100,000 writes (unlimited reads). Do not write the flash EPROM from within a loop. The flash EPROM should be written to only in response to a human request for each write.

Follow these steps to return to Program Mode.

1. Disconnect power from the PK2500.
2. Reinstall the Run/Program jumper on header H2. Refer to Figure 4-2 for the jumper location.
3. Reapply power to the PK2500.

Using Digital Inputs/Outputs

Protected Digital Inputs

The PK2500 provides up to 16 protected digital inputs. These inputs are designed as logical data inputs, returning either a Boolean 1 (ON) or 0 (OFF). The inputs accept voltages in the range of -20 V DC to +24 V DC, with a logic threshold of 2.5 V DC. This means that a protected digital input returns a 0 (OFF) if the input voltage is below 2.5 V DC, and it returns a 1 (ON) if the input voltage is above 2.5 V DC.

Protected digital inputs can be used with +5 V DC CMOS or TTL compatible hardware drivers and sensors. This compatibility allows a system to interface directly with other electronic hardware such as peripheral controllers and various mechanical switches, including relay contacts.

The PK2500 also provides two level-sensitive interrupts that are wired in parallel with protected inputs IN-06 and IN-07. The level-sensitive interrupts are software-assignable.



See Chapter 5, “Software Reference,” for additional details on level-sensitive interrupts.



Appendix B, “Specifications,” provides complete specifications for the PK2500’s protected digital inputs.

How to Read the Inputs

The following Dynamic C software driver reads the status of a specified protected digital input.

```
• int eioBrdDI( unsigned chanNum )
```

Reads the state of an input channel.

PARAMETER: **chanNum** must be a number ranging from 0 (for IN-00) through 15 (for IN-15).

RETURN VALUE:

- 0 if and only if the input channel reads low.
- 1 if and only if the input channel reads high.
- -1 if and only if **chanNum** is out of range (**eioErrorCode** is bit-ored with **EIO_NODEV**).

The sample input demonstration program **DI.C** in the **SAMPLES\PK25xx** subdirectory illustrates the use of the **eiobrdDI** driver.


DI.C

```
#use eziopk25.lib // Use the PK2500 I/O
// library
#define INPUT0 0 // Assign INPUT0 protected
// digital input IN-00

main() {
    int I; // Create integer I
    eiobrdInit(0); // Initialize the PK2500
    I = eiobrdDI(INPUT0); // Assign integer I the
// status of INPUT0
    printf("IN-00 Status: %d\n", I);
// Print status of IN-00
// to the STDIO window
A → I = eiobrdDI(INPUT0); // Assign integer I the
// status of INPUT0
    printf("IN-00 Status: %d\n", I);
// Print status of IN-00
// to the STDIO window
B → } // End of program
```

The following steps explain how to use the sample input demonstration program.

1. Open the sample program.
2. Compile the program by pressing **F3** or by choosing **Compile** from the **Compile** menu.
3. Connect a wire from pin 1, header J1, on the PK2500 (+DC) to pin 6, header J4 (IN-00). This connection provides a logic level 1 at IN-00.



Check to make sure that +DC does not exceed +24 V, the limit on the protected digital inputs, before completing Step 4. The power supply included with the Developer's Kit may exceed 24 V.

4. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **A**. At this point, the Dynamic C **STDIO** window opens and displays the status of IN-00. The status of IN-00 should be 1. If the status is not 1, or the **STDIO** window did not open, check the hardware connections and the program.



The PK2500's protected inputs are factory-configured to be pull-up. The inputs then return a logic level of 1 when not connected to ground.

5. Remove the wire from pin 1, header J1 (+DC), and connect this end to pin 3, header J1 (GND). Leave the end of the wire at pin 6, header J4 (IN-01) connected.
6. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **B**. At this point, the Dynamic C **STDIO** window opens and displays the status of IN-00. The status of IN-00 should be 0. If the status is not 0, or the **STDIO** window did not open, check the hardware connections and the program.
7. Continue to press **F8** until the program terminates.
8. Repeat steps 3 through 7 as desired for the other protected inputs.

High-Current Outputs

The PK2500 provides up to 12 high-current driver outputs in two banks of six. Each group of six outputs is factory-configured with “sinking” drivers, and can optionally be configured with “sourcing” drivers. By configuring one bank as sourcing and the other as sinking, it is possible to create push-pull or bidirectional drivers to control loads such as small reversible DC motors. Each bank of high-current driver outputs can provide up to 500 mA, with a maximum output of 75 mA per output channel.



Appendix D, “Sinking vs. Sourcing Drivers,” provides more information on sinking and sourcing drivers.

How to Use the Outputs

The following Dynamic C software turns a specified high-current driver ON or OFF.

- `int eioBrdDO(unsigned chanNum, char state)`

Changes the state of an output channel.

PARAMETERS: `chanNum` must range from 0 (for OUT-00) through 11 (for OUT-11).

`state` is 0 if and only if the corresponding output is to be disabled (OFF); 1 if and only if the corresponding output is to be enabled (ON)

RETURN VALUE:

- 0 if and only if `chanNum` is within range.
- -1 if and only if `chanNum` is out of range (`eioErrorCode` is bit-ored with `EIO_NODEV`).

The sample input demonstration program `DO.C` in the `SAMPLES\PK25xx` subdirectory illustrates the use of the `eioBrdDO` driver.

DO.C

```
#use eziopk25.lib // Use the PK2500 I/O
// library
#define OUTPUT0 0 // Assign OUTPUT0 high-
// current output OUT-00
#define ON 1 // Assign ON the logic
// value 1
#define OFF 0 // Assign OFF the logic
// value 0

main() {
    eioBrdInit(0); // initialize the PK2500
    eioBrdDO(OUTPUT0,OFF); // Turn-off OUT-00
A → eioBrdDO(OUTPUT0,ON); // Turn-on OUT-00
B → eioBrdDO(OUTPUT0,OFF); // Turn off OUT-00
C → } // End of program
```

The following steps explain how to use the sample output demonstration program.

1. Open the sample program.
2. Compile the program by pressing **F3** or by choosing **Compile** from the **Compile** menu.
3. Connect a wire from pin 1, header J1, on the PK2500 (+DC) to pin 2, header J1 (KA). Connect a device such as a voltmeter between pin 4, header J1 (OUT-00) and pin 2 (KA), as shown in Figure 4-3, to monitor the status of the driver.

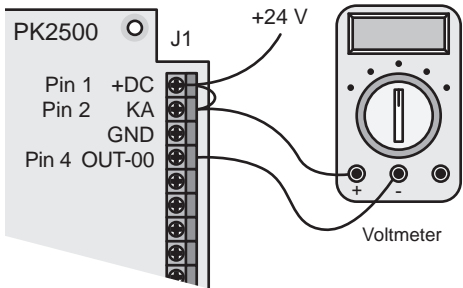


Figure 4-3. OUT-00 Status Example



Exercise caution when connecting any device to a high-current driver. Do not exceed the voltage and current limitations of the high-current driver. Appendix B contains complete specifications.

4. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **A**. At this point, the device should be “OFF.” The voltmeter will read approximately 0 V because the output is not pulled to ground. If the device is not OFF, check the hardware connections and the program.
5. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **B**. At this point, the device should be “ON.” The voltmeter will read approximately +24 V (+DC) because the output is pulled to ground. If the device is not ON, check the hardware connections and the program.
6. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **C**. At this point, the device should be “OFF” again (the voltmeter will read approximately 0 V). If the device is not OFF, check the hardware connections and the program.
7. Continue to press **F8** until the program terminates.
8. Repeat steps 3 through 7 as desired for the other outputs.

Serial Communication

The PK2500 provides two serial communication ports that can be configured as RS-232 and/or RS-485.



See Chapter 3, “Input/Output Configuration,” for further details on serial channel configurations.

RS-232 Communication

RS-232 is an asynchronous serial communication protocol that is full-duplex (simultaneous bidirectional data transfer). The RS-232 ports and the Dynamic C software allow the PK2500 to communicate with other computers or controllers. A modem allows remote communication (including remote downloading) by using the X-modem protocol. RS-232 software drivers can be found in the Dynamic C `SAMPLES\AASC` subdirectory.



Refer to the Dynamic C manuals for additional information on remote downloading.

Tip

The optional Serial Interface Board 2 leaves both serial ports available to the application during software development. A special cable has to be made to access J2 if both RS-232 ports are needed.

RS-232 Connector Pinouts

The 6-pin RJ-12 modular phone jack (J2) facilitates all RS-232 connections. Table 4-2 lists the pin assignments for connector J2.

Table 4-2. J2 Pin Assignments

J2 Pin	Handshaking (Configuration II)	No Handshaking (Configuration I or III)
1	RTS RS-232 (0)	Transmit RS-232 (1)
2	GND	GND
3	Transmit RS-232 (0)	Transmit RS-232 (0)
4	Receive RS-232 (0)	Receive RS-232 (0)
5	CTS RS-232 (0)	Receive RS-232 (1)
6	+5 V regulated	+5 V regulated



Do not pull more than 150 mA from pin 6 of J2.

RS-485 Network

The PK2500 can be configured to provide one channel of RS-485 communication. RS-485 is an asynchronous multidrop half-duplex standard that provides multidrop networking with maximum cable lengths up to 1000 m.

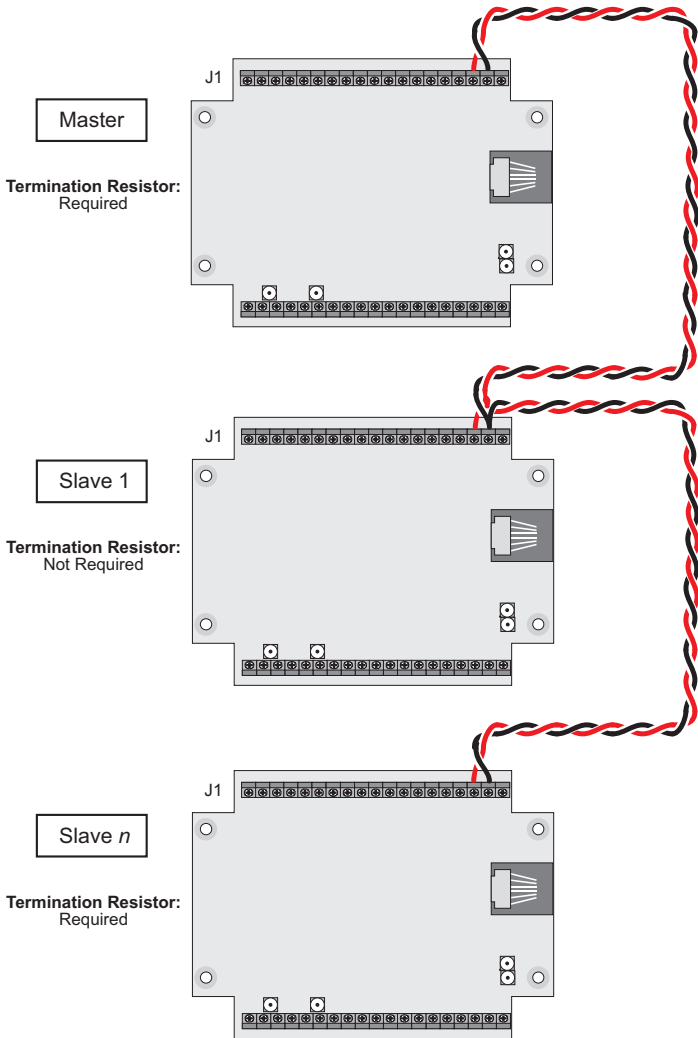


Figure 4-4. RS-485 Multidrop Network



The RS-485 drivers support up to 32 nodes. The transmission bandwidth may be reduced if more than 32 nodes are added. Contact Z-World Technical Support for assistance with large-scale network design.

Dynamic C provides library functions for master-slave two-wire half-duplex RS-485 9-bit binary communications.

This RS-485 hardware standard supports up to 32 controllers on one network. The software supports one master unit, plus up to 255 slave units (which may consist of any combination of Z-World controllers that support the RS-485 protocol).

The resulting multidrop network (shown in Figure 4-4) can span up to a kilometer, facilitating the design of a robust distributive control system.

Follow these steps to configure a multidrop network.

1. Configure pins 17 and 18 on header J1 for RS-485 communications, as outlined in the “Configuring Serial Communications” section in Chapter 3.
2. Connect RS-485+ to RS-485+ and RS-485- to RS-485- on all networked controllers, using single twisted-pair wires. Use Figure 4-4 for reference.



Refer to the Dynamic C manuals for additional information on master-slave networking.

Termination and Bias Resistors

Termination and bias resistors are required in a multidrop network to minimize reflections (echoing) and to keep the network line active when it is in an idle state.

Bias resistors are required in most cases on the master controller to keep the network line reliable. The PK2500 is factory-configured with 10 k Ω bias resistors installed.

An external termination resistor is not required. A jumper on header H5 determines whether the onboard termination resistor is connected to the network. The termination resistor is needed only at the physical ends of the network, as shown in Figure 4-4. The resistor is disconnected by removing the jumper. Figure 4-5 summarizes the jumper configuration.

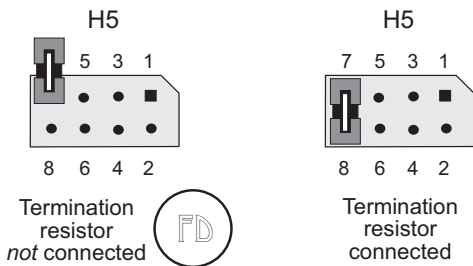


Figure 4-5. PK2500 Jumper Settings on Header H5 for Termination Resistor

Relay Outputs

The PK2500 has two relays to drive external loads. The two contacts for Relay 0 are available at pins 16 and 17 on header J4, while those for Relay 1 are available at pins 18 and 19 on header J4. Both sets of contacts are normally open. The relays are rated for a 60 W load, for example, 2 A at 30 V DC or 0.5 A at 125 V AC. The maximum switching voltage is 125 V.



For CE compliance, the maximum relay switching voltage is less than 50 V AC or 75 V DC.

Two LEDs are provided to provide a visual aid for development or for field diagnostics. LED D3 turns on when Relay 1 is activated. LED D2 does the same for Relay 0.

Figure 4-6 shows the relay outputs.

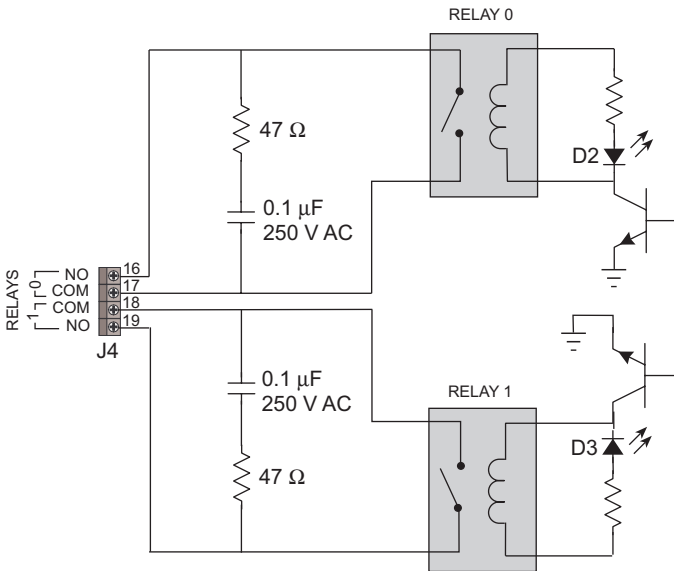


Figure 4-6. Relay Outputs

An onboard snubber network is provided for each relay to attenuate switching transients that develop when inductive loads are driven. Each network consists of a 47 Ω resistor in series with a 0.1 μF capacitor. It is remotely possible for some switching transients to be large enough to cause disruptive interference to the PK2500 logic. This depends to a large degree on the size and reactance of the load. This should not be a problem because the relays are only rated for 60 W resistive loads. Inductive loads should be derated. External transient suppressors, such as metal oxide

varistors (MOVs) may be added in parallel with the relay contacts if there are problems despite the built-in snubber circuitry. MOVs come in a range of protective-voltage ratings. Select one with a voltage rating slightly higher than that of the load.

Figure 4-7 shows the MOVs installed across the relay outputs.

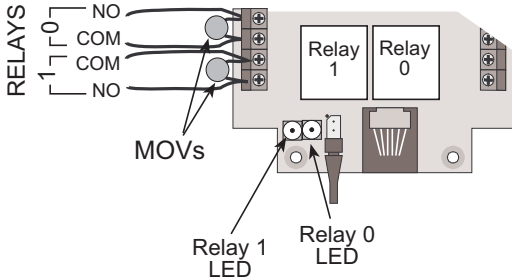


Figure 4-7. Installation of Metal Oxide Varistors to Suppress Transients



Use heat-shrink tubing over the exposed portions of the MOV leads, especially if high voltages (over 48 V) are being switched.

How to Use the Relay Outputs

The following Dynamic C software turns a specified relay ON or OFF.

- `int eioBrdRelay(unsigned chan, char onOff)`

Changes the state (whether the relays are energized) of the two relays on the PK2500.

PARAMETERS: `chan` identifies which relay to switch—0 for Relay 0 or 1 for Relay 1.

`onOff` specifies whether the relay should be energized (non-zero) or de-energized (zero).

RETURN VALUE:

- 0 if the operation was successful.
- -1 if and only if `chan` is out of range (`eioErrorCode` is bit-ored with `EIO_NODEV`).

The sample relay output demonstration program **REL.C** in the **SAMPLES\PK25xx** subdirectory illustrates the use of the **eioBrdRelay** driver.

REL.C

```
#use eziopk25.lib      // Use the PK2500 I/O
                        // library
#define OUTPUT0      0 // Assign OUTPUT0
                        // relay output REL0
#define ON           1 // Assign ON the logic
                        // value 1
#define OFF          0 // Assign OFF the logic
                        // value 0

main() {
    eioBrdInit(0);      // initialize the PK2500
    eioBrdRelay(OUTPUT0,OFF); // Turn off REL0
A → eioBrdRelay(OUTPUT0,ON); // Turn on REL0
B → eioBrdRelay(OUTPUT0,OFF); // Turn off REL0
C → }                  // End of program
```

The following steps explain how to use the Relay Output Demonstration Program.

1. Open the sample program.
2. Compile the program by pressing **F3** or by choosing **Compile** from the **Compile** menu.
3. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **A**. At this point, the relay should be “OFF.” If the relay is not OFF, check the hardware connections and the program.
4. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **B**. At this point, the relay should be “ON,” and LED D2 should flash. If the relay is not ON, check the hardware connections and the program.
5. Use the Dynamic C command **F8** (run/step over) to single-step through the program to the line marked **C**. At this point, the relay should be “OFF” again. If the relay is not OFF, check the hardware connections and the program.
6. Continue to press **F8** until the program terminates.
7. Repeat steps 3 through 6 as desired for the other relay output.

Analog-to-Digital Converter Inputs

The PK2500 has four analog input channels that are brought out on header J4. The analog inputs are designated AIN0 to AIN3, and are listed in Table 4-3.

Table 4-3. Analog Input Channel Numbers

Header J4 Pin No.	Signal	Dynamic C Channel No.
1	AIN0	0
2	AIN1	1
3	AIN2	2
4	AIN3	3
5	GND	Analog Ground



Use pin 5 on header J4 for analog ground only. Do not use this pin as a ground for any other signals.

The analog input channels can be configured for a wide range of input voltages. The default input range of the analog inputs is 0 V to 10 V. The analog inputs can be configured for almost any input range by replacing two resistors in the input amplifier circuit.

Each analog channel consists of an inverting amplifier referenced to a user-defined offset voltage, as shown in Figure 4-8.

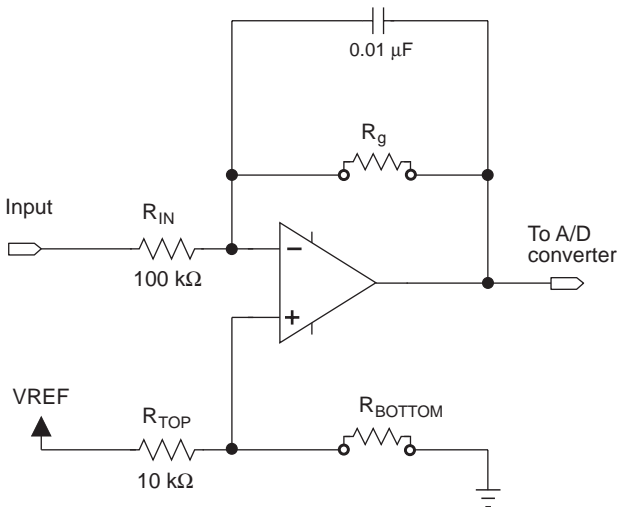


Figure 4-8. Analog Input Amplifier

Table 4-4 shows the analog input amplifier components.

Table 4-4. Analog Input Component References

Channel	R_{IN}	R_g	R_{TOP}	R_{BOTTOM}	CAP
AIN0	R3	R2	R5	R4	C1
AIN1	R17	R16	R14	R15	C21
AIN2	R22	R23	R18	R19	C25
AIN3	R25	R24	R20	R21	C26

Changing the values of R_g and R_{BOTTOM} sets the gain and offset of the channel. Figure 4-9 shows the locations of the R_g and R_{BOTTOM} resistors.

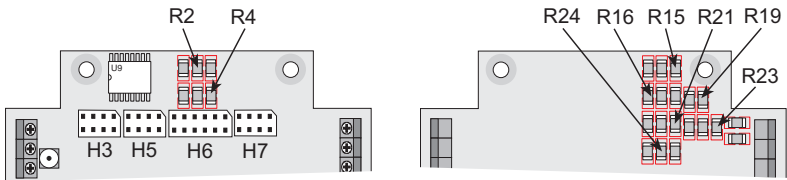


Figure 4-9. Locations of R_g and R_{BOTTOM} Resistors

Table 4-5 shows resistor values for some common analog input voltage ranges.

Table 4-5. Resistor Values for Common Input Ranges

Channel Input Range (V)	R_g (k Ω)	R_{BOTTOM} (k Ω)
-10.0 to +10	11.8	8.06
- 5.0 to + 5.0	23.7	6.65
- 2.5 to + 2.5	47.5	4.99
- 2.0 to + 2.0	59.0	4.53
- 1.0 to + 1.0	118	2.87
- 0.5 to + 0.5	237	1.69
- 0.25 to + 0.25	464	0.953
- 0.10 to + 0.10	1180	0.392
0 to +10.0	23.7	39.2
0 to + 5.0	46.4	19.6
0 to + 2.5	93.1	10.0
0 to + 1.0	226	4.02
+1 to + 2.0	237	13.3
+2 to + 7.0	47.5	140.0
+0.020 to + 0.100	3010	0.402

If the application requires an input voltage range not shown in Table 4-5, the correct values of R_g and R_{BOTTOM} can be determined using the formulas presented in the next section.

R_g and R_{BOTTOM} are both 1206-size surface-mount technology (SMT) resistors.



The analog input resistors and capacitors are surface-mounted. Use proper SMT techniques (see Z-World Technical Note 106, *Soldering and Desoldering Surface-Mount Chip Components*) to remove and replace these parts.

Scaling Input Range

Once the input range has been determined, the appropriate resistor values must be selected. The steps below describe this process.

1. The first step in setting up the channel is to choose a gain resistor. Use Equation (4-1) to determine the value of the gain resistor.

$$R_g = \frac{2.5 \times 10^5}{V_{IN_{max}} - V_{IN_{min}}} \quad (4-1)$$

2. Next, select R_{BOTTOM} using Equation (4-2).

$$R_{BOTTOM} = \frac{R_g \cdot V_{IN_{max}} \cdot 10^4}{R_g \cdot (2.5 - V_{IN_{max}}) + (2.5 \times 10^5)} \quad (4-2)$$

3. Select the appropriate resistor values. Standard resistor values can be found in many electronics references and catalogs. Using 1% resistors will give better accuracy and a greater number of choices than using 5% resistors.

- For R_g , select the next standard value less than the standard value closest to the computed value. Choosing a lower value helps insure that the input signal does not exceed the 2.5 V reference voltage for the ADC.
- For R_{BOTTOM} , select the nearest standard value.

4. To verify that the calculated values provide the correct gain and offset, plug $V_{IN_{min}}$ and $V_{IN_{max}}$ into Equation (4-3).

$$V_{OUT} = \left(\frac{R_{BOTTOM}}{R_{BOTTOM} + 10^4} \cdot 2.5 - V_{IN} \right) \cdot \frac{R_g}{10^5} + \frac{R_{BOTTOM}}{R_{BOTTOM} + 10^4} \cdot 2.5 \quad (4-3)$$

$V_{IN_{min}}$ should yield a positive V_{OUT} just less (within 50 mV) than 2.5 V.
 $V_{IN_{max}}$ should yield a positive V_{OUT} just greater (within 50 mV) than zero.

5. Test the circuit to verify it works properly.

Example

Given a sensor that has an output of -1 to 2 V, $V_{IN_{min}}$ and $V_{IN_{max}}$ are as follows.

$$V_{IN_{max}} = 2 \text{ V}, V_{IN_{min}} = -1 \text{ V}$$

1. Determine the value of the gain resistor using Equation (4-1).

$$R_g = \frac{2.5 \times 10^5}{(2) - (-1)} = 83.3 \text{ k}\Omega \text{ (ideal value).}$$

2. Select R_{BOTTOM} using Equation (4-2).

$$R_{BOTTOM} = \frac{83.3 \times 10^3 \cdot 2 \cdot 10^4}{(8.33 \times 10^3) \cdot (2.5 - (2)) + (2.5) \cdot (10^5)} = 5.71 \text{ k}\Omega \text{ (ideal value).}$$

3. Select resistor values.

- The next lower standard 1% value for R_g is 80.6 k Ω .
- The closest standard 1% value for R_{BOTTOM} is 5.76 k Ω .

4. Now, check values by plugging V_{INmin} and V_{INmax} into Equation (4-3).

- Check V_{OUT} for V_{INmin} .

$$V_{OUT} = \left(\frac{5.76 \times 10^3}{5.76 \times 10^3 + 10^4} \cdot 2.5 - (-1) \right) \cdot \frac{80.6 \times 10^3}{10^4} \\ + \frac{5.76 \times 10^3}{5.76 \times 10^3 + 10^4} \cdot 2.5 \\ = 2.456 \text{ V.}$$

This value of V_{OUT} is within 50 mV of 2.5 V using V_{INmin} .

- Check V_{OUT} for V_{INmax} .

$$V_{OUT} = \left(\frac{5.76 \times 10^3}{5.76 \times 10^3 + 10^4} \cdot 2.5 - (2) \right) \cdot \frac{80.6 \times 10^3}{10^4} \\ + \frac{5.76 \times 10^3}{5.76 \times 10^3 + 10^4} \cdot 2.5 \\ = 0.038 \text{ V.}$$

This value of V_{OUT} is within 50 mV of 0 V using V_{INmax} .

Both V_{OUT} numbers fall within the acceptable output range.

In this example, with $R_g = 80.6 \text{ k}\Omega$ and $R_{BOTTOM} = 5.76 \text{ k}\Omega$, the analog channel will accept inputs from -1 V to 2 V. The amplifier output is then in the range from 0 V to 2.5 V, with a little margin (~30 mV) for system error (for example, ADC or op-amp offset).

Theory of Operation

This section presents a complete analysis of the analog input circuit (see Figure 4-8).

The initial first-order approximation of the op amp assumes the following:

- Infinite open-loop gain,
- Zero output impedance,
- Zero voltage offset,
- Infinite input impedance,
- Zero input bias currents, and
- Noiseless components.

The V_{OFF} Voltage Divider

The voltage divider formed by R_{TOP} and R_{BOTTOM} provides an offset voltage for the amplifier. The offset voltage V_{OFF} is given by Equation (4-4).

$$V_{OFF} = \frac{R_{BOTTOM}}{R_{BOTTOM} + R_{TOP}} \cdot V_{REF} \quad (4-4)$$

V_{REF} for the PK2500 is 2.5 V.

DC Gain

Now, examine the DC gain of the circuit.

An amplifier in a negative feedback topology will force the error between the amplifier's inputs to zero. This implies the following:

$$V_{(INVERTING)} = V_{(NONINVERTING)} = V_{OFF} \quad .$$

Since there is infinite impedance at the op amp's inputs, the current through R_{IN} must equal the current through R_g . The current through R_{IN} is determined by Equation (4-5).

$$I_{R_{IN}} = \frac{V_{IN} - V_{OFF}}{R_{IN}} \quad (4-5)$$

The current through R_g , I_{R_g} , is

$$I_{R_g} = \frac{V_{OFF} - V_{OUT}}{R_g} \quad . \quad (4-6)$$

Setting Equation (4-5) and Equation (4-6) equal and solving for V_{OUT} yields the following:

$$V_{OUT} = \left(\frac{V_{OFF} - V_{IN}}{R_{IN}} \right) \cdot R_g + V_{OFF} \quad . \quad (4-7)$$

When V_{OFF} is zero, the circuit scales V_{IN} by a factor of $-R_g/R_{IN}$. This is the DC gain. The DC gain can be determined by Equation (4-8).

$$g = \frac{-R_g}{R_{IN}} \Rightarrow |g| = \frac{R_g}{R_{IN}} \quad (4-8)$$

This result agrees with the gain of a classic op amp inverting amplifier.

Given a desired input range, it is possible to compute the required circuit gain.

The amplifier needs to map the input voltage range to the $0-V_{REF}$ input range of the ADC. Thus

$$g = \frac{V_{REF}}{V_{IN_{max}} - V_{IN_{min}}} \quad (4-9)$$

The PK2500 has $V_{REF} = 2.5$ V and R_{IN} is shipped as 10 k Ω .

Once the gain has been computed, Equation (4-8) can be used to compute R_g .

Finding V_{OFF}

Once the gain is known, V_{OFF} required to center the output in the range between 0 V and V_{REF} can be determined.

Examine V_{OFF} when V_{IN} is at minimum and maximum.

First, take the case of a maximum V_{IN} . Because the circuit is an inverting amplifier, $V_{IN_{max}}$ must map V_{OUT} to zero.

Start with Equation (4-7) and substitute $V_{OUT} = 0$ and $V_{IN} = V_{IN_{max}}$ to get

$$\begin{aligned} 0 &= \left(\frac{V_{OFF} - V_{IN_{max}}}{R_{IN}} \right) \cdot R_g + V_{OFF} \\ &= (V_{OFF} - V_{IN_{max}}) \cdot \frac{R_g}{R_{IN}} + V_{OFF} \end{aligned} \quad (4-10)$$

Now, simplify using Equation (4-8). Substitute Equation (4-8) into Equation (4-10).

$$\begin{aligned} 0 &= (V_{OFF} - V_{IN_{max}}) \cdot (-g) + V_{OFF} \\ &= (V_{IN_{max}} - V_{OFF}) \cdot g + V_{OFF} \end{aligned} \quad (4-11)$$

Then,

$$V_{OFF} = V_{IN_{max}} \cdot \frac{g}{g+1} \quad (4-12)$$

Do the same thing for $V_{IN} = V_{INmin}$. When V_{INmin} is presented at V_{IN} , $V_{OUT} = V_{REF}$ has to be true. Start with Equation (4-7) and substitute $V_{OUT} = V_{REF}$ and $V_{IN} = V_{INmin}$ to get

$$\begin{aligned}
 V_{REF} &= \left(\frac{V_{OFF} - V_{INmin}}{R_{IN}} \right) \cdot R_g + V_{OFF} \\
 &= (V_{OFF} - V_{INmin}) \cdot \frac{R_g}{R_{IN}} + V_{OFF} \\
 &= (V_{OFF} - V_{INmin}) \cdot (-g) + V_{OFF} \\
 &= (V_{INmin} - V_{OFF}) \cdot g + V_{OFF} \\
 &= V_{INmin} \cdot g + V_{OFF}(1 - g) \quad .
 \end{aligned}
 \tag{4-13}$$

Rearrange and solve for V_{OFF} .

$$V_{OFF} = \frac{V_{INmin} \cdot (g) - V_{REF}}{(g - 1)}
 \tag{4-14}$$

Either Equation (4-12) or Equation (4-14) may be used to calculate V_{OFF} .

Practical Considerations

Resistors are available only in discrete standard values. Once ideal values are computed, a standard value must be selected. There are more 1% resistor values to choose from than there are 5% resistor values. Also, 1% resistors have a lower temperature drift. There is only a slight difference in cost between 1% and 5% resistors.

Resistors over 3 M Ω should be avoided. Two reasons to avoid large resistor values are circuit noise susceptibility and part availability. Reducing R_{IN} or adding an external preamp are alternative methods of increasing gain if a large DC gain is needed.

The op amp in the PK2500 has a ± 7.5 mV maximum offset voltage (at 25°C). This offset is multiplied by the D.C. gain and added to V_{OUT} . This means if the DC gain is 10, V_{OUT} may have an offset in it of up to 75 mV.

The input bias currents of the op amp will also produce error voltages at the inputs that get multiplied by the DC gain, and will show up as an offset in V_{OUT} . Fortunately, the input bias currents are very low, of the order of picoamperes, and so this effect should be negligible.

To avoid these offsets pushing V_{OUT} beyond the 0– V_{REF} range of the ADC, select R_g to be a smaller standard value than computed. This will sacrifice some dynamic range of the ADC for improved reliability.

Selecting R_{BOTTOM} is a matter of picking the standard value closest to the computed value.

Input Impedance

The input impedance looking into the circuit from V_{IN} is just R_{IN} . Note also that R_{IN} is connected to the inverting input, which is maintained (by the op amp's negative feedback) at V_{OFF} .

Gain can be increased by sacrificing input impedance. A fixed value of R_g will produce a larger DC gain if R_{IN} is reduced. However, a smaller R_{IN} will require the source of V_{IN} (often a transducer) to provide additional current, as shown in Equation (4-15).

$$I_{R_{IN}} = \frac{V_{IN} - V_{OFF}}{R_{IN}} \quad (4-15)$$

Frequency Response

The capacitor in the feedback loop fixes a pole as shown in Equation (4-16). F_c , the 3 dB point for the single-pole filter is in hertz, and C is capacitance.

$$F_c = \frac{1}{2\pi \cdot R_g \cdot C} \quad (4-16)$$

The filter will roll off at the 20 dB decade after F_c .

This low-pass filter helps eliminate noise in the channel. The pole should be set as low as possible for the application. The standard capacitor shipped is 0.01 μ F.

How to Use the Analog-to-Digital Converter

The best way to use the A/D converter is with the Z-World Dynamic C drivers. This helps to ensure that the code will be compatible with future versions of the PK2500 as well as other Z-World products.

Using the Analog Inputs

The factory calibrates each PK2500, storing each unit's individual zero offset and actual gain for each channel in simulated EEPROM. The library function `eioBrdAI` uses these calibration values to provide adjusted readings for the analog inputs.

- `float eioBrdAI(unsigned int eioAddr)`

Reads an input and performs analog-to-digital conversion.

PARAMETERS: `eioAddr` specifies an input of 0 to 3 or 16 to 19 to be read. `eioAddr` values 0 through 3 represent analog inputs 0 through 3 and cause the function to return the voltage read at an input. `eioAddr` values 16 through 19 also represent analog inputs 0 through 3, but cause the function to return a 12-bit raw data value for the analog input.

RETURN VALUE:

- Voltage read for `eioAddr` values 0 through 3, if the read is successful.
 - 12-bit raw data value read from the A/D converter for `eioAddr` values 16 through 19, if the read is successful.
 - Sets `eioErrorCode` if `eioAddr` is out of range.
- `int eioBrdACalib(int chanNum, int d1, int d2, float f1, float f2)`

Sets up the calibration constants needed by `eioBrdAI` when called with `chanNum` equal to 0.

The function computes the calibration coefficients and stores them in reserved locations in nonvolatile memory. The function `eioBrdInit` loads these constants from nonvolatile memory.

PARAMETERS: `chanNum` must range from 0 (for AIN0) through 3 (for AIN3).

`d1` is the raw, digital reading corresponding to the applied analog resistance `f1`.

`d2` is the raw, digital reading corresponding to the applied analog resistance `f2`.

The sample input demonstration program **AI.C** in the **SAMPLES\PK25xx** subdirectory illustrates the use of the **eioBrdAI** driver.

AI.C

```
#use eziopk25.lib
#define INPUTCHAN 0
main() {
    float raw, analog;
    eioBrdInit(0); // initialize the I/O driver
    eioErrorCode = 0; // clear error flag
    raw = eioBrdAI(INPUTCHAN+16);
                                // read the raw chan.
    /* read channel, scale with calibration constants */
    eioErrorCode = 0; // clear error flag
    analog = eioBrdAI(INPUTCHAN);
    if (eioErrorCode & EIO_NODEV)
    {
        printf("analog input channel %d doesn't
            exist!\n", INPUTCHAN);
    }
    else
    {
        printf("analog input channel %d reads 0x%04x,
            interpreted to %f\n", INPUTCHAN,
            (int)raw, analog);
    }
}
```

Using the A/D Voltage Reference

The jumpers on header H5 are used to configure pin 4 of header J4 to provide a buffered copy of the internal reference voltage used by the A/D converter. This is useful in applications such as ratiometric measurements if the sensors need to track the same voltage reference used by the A/D converter.



The Channel 3 op-amp serves as the buffer. Do not draw more than 2 mA to 3 mA since the op-amp's current drive is limited.



Chapter 3, "Input/Output Configuration," provides the exact jumper configurations for header H5 to serve as either an A/D converter input or and A/D voltage reference.

Additional Features

This section provides information on the PK2500's additional features including pulse-width modulated outputs, user-programmable LEDs, the real-time clock, the power supervisor, and a +5 V output.

Pulse-Width Modulated (PWM) Outputs

The 12 high-current outputs on the PK2500 are driven by two driver chips in two banks of six outputs each. The PWM feature is available on up to six of the outputs in either group at any one time.

The supplied software provides two levels of support. The first level provides easy-to-use fixed PWM functions for only four of the outputs. The periods of the PWM signals are fixed at 13.3 ms (75 Hz), with a resolution of 256 divisions per period (8-bit resolution). Dynamic C consumes about 8% of the PK2500's processing power when used to generate PWM signals. The second PWM support level allows custom PWM functions to be created for six of the outputs.



Serial-communication baud rates may be affected when PWM functions are used because the microprocessor's functions may get overloaded. Serial data rates become limited and fixed at 4800 bps for Serial Port 1. Be sure to reset the Dynamic C baud rate to 4800 bps.

Contact Z-World Technical Support at (530)757-3737 for assistance with PWM functions.



See Chapter 5, "Software Reference," for advanced PWM programming information.

How to Use the PWM Feature

The PK2500 can produce fixed-frequency, fixed-phase, variable-duty-cycle square waves from up to six of its high-current outputs. This section first presents a simple, easy-to-use PWM function that drives only four of the PK2500's outputs. A more complex set of functions that require more in-depth understanding of DMA and PWM generation is presented later.

Figure 4-10 and Figure 4-11 provide PWM transition and DMA timing diagrams.

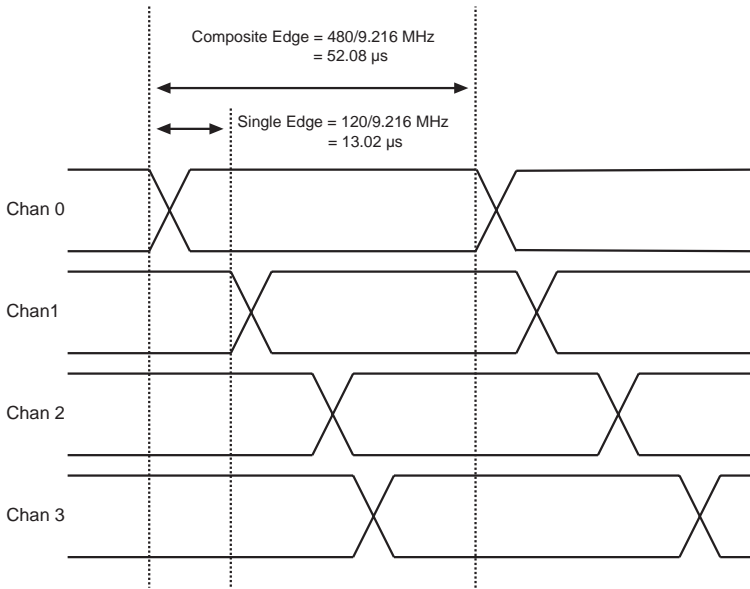


Figure 4-10. Transition Timing

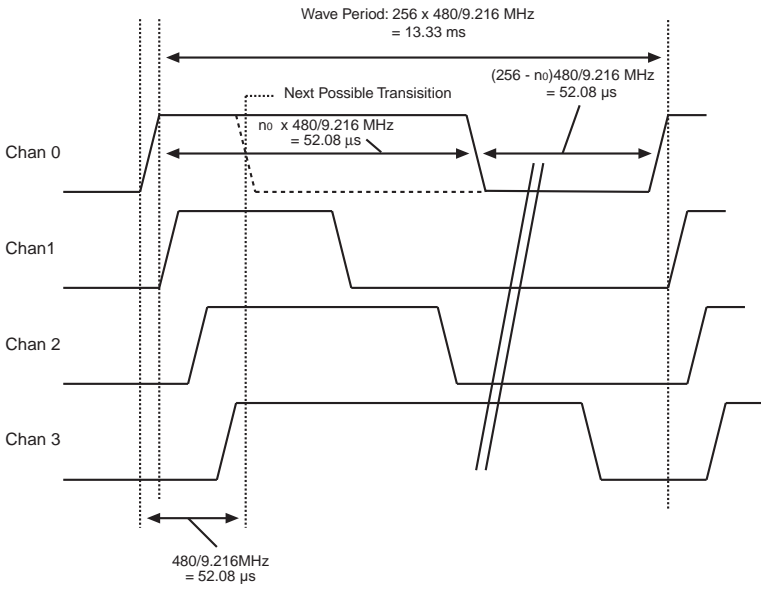


Figure 4-11. DMA Timing

- **int eioBrdAO(unsigned chanNum, unsigned state)**

Specifies the duty cycle for a particular output channel.

- **chanNum** is a number ranging from 0 (for OUT-00) to 3 (for OUT-03).
- **state** is a placeholder for a number ranging from 0 (to turn off the channel) to 256 (to turn on the channel, 100% duty cycle). The duty cycle is **state**/256 (for example, 128 for 50% duty cycle, 64 for 25% duty cycle).

The function produces PWM square waves on outputs OUT-00, OUT-01, OUT-02, and OUT-03. Notice that each square wave's period is exactly 1024 "divisions." One division equals 120 clock cycles ($120/9.216 \text{ MHz} = 13.02 \mu\text{s}$) for the PWM function. Consequently, the period of each square wave is $1024 \times 13.02 \mu\text{s} = 13.33 \text{ ms}$.

Notice also that the square waves are displaced slightly from each other in phase. That is, OUT-01's output starts and ends one division after OUT-00's, OUT-02's starts one division after OUT-01's, and OUT-03's starts one division after OUT-02's. As a result, a change to one particular channel is possibly only every four divisions even though the period of each wave is 1024 divisions. Therefore, the resolution of the transition edge in the wave is $1/256$.



The PWM function may appear to be simple, but be aware of the side effects. The function uses the Z180's built-in DMA hardware. This limits the baud rate on Z180 Serial Port 1 to 4800 bps, and the Z180 effectively runs 8% slower.

Make sure any application calling this PWM function also calls **_eioBrdAORf** every 25 ms to refresh the driver period.

Contact Z-World Technical Support at (530)757-3737 for assistance with PWM functions.

The following sample program, **AO1.C**, in the **SAMPLES\PK25xx** subdirectory ramps the PWM output from OUT-00 up and down.

AO1.C

```
#use eziopk25.lib      // pk2500 specific defs
#define OUT_CHAN 0     // define output channel

main() {
    auto unsigned dutyCycle;
    auto int sign;
    eioBrdInit(0);     // initialize general I/O
    _eioSetupAO1st(); // initialize PWM
    dutyCycle = 0;     // duty cycle starts at 0
    sign = 1;         // ramp up
    while (1) {       // do this forever
        eioBrdAO(OUT_CHAN,dutyCycle); // change cycle
        if (_eioBrdAORf()==-1) break; // refresh OK?
        if (dutyCycle == 256) sign = -1; // reverse
        else if (dutyCycle == 0) sign = 1; // reverse
        dutyCycle = dutyCycle + sign; // ramp
        hitwd();       // hit watchdog
    }
    printf("AO refresh failed\n");
}
```

User-Programmable LEDs

The PK2500 provides two clearly visible surface-mounted LEDs, **RUN** (D5) and **USER** (D4). The program can control both LEDs. The **USER** LED indicates the various operating modes during program development—it illuminates steadily to indicate power-on and that the Z-World factory default BIOS is functioning.

The following software drivers can be used to turn the LEDs ON or OFF.

- **outport(0x4141,1)** • **outport(0x4142,1)**
Turns **RUN** LED ON. Turns **USER** LED ON.
- **outport(0x4141,0)** • **outport(0x4142,0)**
Turns **RUN** LED OFF. Turns **USER** LED OFF.

Real-Time Clock (RTC)

The PK2500's real-time clock maintains the current time and date, accounts for the number of days in different months, and accounts for leap years. A backup battery keeps the real-time clock running when power is removed.

The Dynamic C function library `DRIVERS.LIB` provides these RTC functions.

- `tm_rd`
Reads time and date values from the RTC.
- `tm_wr`
Writes time and date values to the RTC.

Power Supervisor



The *Dynamic C Function Reference* manual provides a complete description of these RTC functions and their associated `tm` data structures.



The real-time clock is unable to update during a read cycle.

Do not use the real-time clock to create small delays in an application. This could lead to a loss of accuracy.

Constant reading of the clock in a tight loop may lead to a loss in accuracy.

Z-World does not recommend using the real-time clock to schedule events.

The PK2500 provides a power supervisor IC that controls the power-on reset function. This function holds the reset line low until VCC rises above the threshold of ~4.75 V.

When VCC falls below this threshold, the supervisor disables the SRAM to prevent writing spurious data. The supervisor also switches the SRAM to battery power when VCC falls below the threshold voltage to preserve the SRAM's data until power is restored.

The supervisor has a watchdog timer that guards against system or software faults. The PK2500's microprocessor will reset if the application's software does not reset the timer at least once every second. The Dynamic C function `hitwd` resets the supervisor's watchdog timer.



Refer to the Dynamic C reference manuals for further information on `hitwd`.

In addition, the supervisor generates a nonmaskable interrupt (NMI) when the unregulated DC input (normally 9 V to 12 V DC) falls below 8.02 V to allow the PK2500's microprocessor time to execute a safe shutdown. Together with the appropriate interrupt service routine and external power supply capacitance, this circuitry will allow the PK2500 to recover from brownouts.

The PK2500 is able to distinguish between a power-on reset and a watchdog reset.



See Appendix C, "Power Management," for further information on power failure provisions.

Tip

Use the `hitwd` function in any program loop that takes longer than one second to execute.

+5 V Output

An onboard switching regulator supplies +5 V derived from the +DC unregulated input voltage. There is spare capacity of approximately 150 mA available for external loads on pin 14 of header J4. The +5 V is also present on header H4, but there are no connectors available at that location.



CHAPTER 5: **SOFTWARE REFERENCE**

Chapter 5 describes the Dynamic C functions that initialize the PK2500 and perform I/O operations. The following sections are included.

- Input/Output Software Drivers
 - Digital Inputs/Outputs
 - Level-Sensitive Interrupts
 - Interrupt Service Routines
- Pulse-Width Modulation Outputs
- Advanced Input/Output Programming
 - Digital Input Addressing Detail
 - Digital Output and Relay Output Addressing Details
 - Analog-to-Digital Converter Addressing Details
 - LED Addressing Details
 - RS-485 Driver Addressing Details
 - PWM Addressing Details
 - PWM Advanced Programming Functions

Input/Output Software Drivers

Dynamic C provides a series of software drivers for controlling the PK2500's inputs/outputs (I/O). These drivers are located in the Dynamic C **EZIOPK25.LIB** library. Include the following line at the start of an application program to access this library.

```
#use eziopk25.lib
```

The **EZIOPK25.LIB** library contains the following functions.

- **void eioBrdInit(int param)**

Initializes the software.

Call this function in the initialization section of a program before using any other functions. Always pass 0 for **param**.

This function does not call **_eioSetupAO1st**. Call **_eioSetupAO1st** separately if the DMA-driven pulse-width modulation output is used.

- **int eioErrorCode**

Represents a global bit-mapped register whose flags reflect error occurrences.

This register is initially set to 0 by **eioBrdInit**. The flag **EIO_NODEV** (the first bit flag) is set in this register if the application tries to access an invalid channel. Note that the other bits in **EIO_NODEV** deal with networked controllers.

Digital Inputs/Outputs

The following digital I/O functions are located in the **EZIOPK25.LIB** library.

- **int eioBrdDI(unsigned chanNum)**

Reads the state of an input channel.

PARAMETER: **chanNum** must be a number ranging from 0 (for IN-00) through 15 (for IN-10).

RETURN VALUE:

- 0 if and only if the input channel reads low.
- 1 if and only if the input channel reads high.
- -1 if and only if **chanNum** is out of range, that is, **chanNum** is greater than 15 (**eioErrorCode** is bit-ored with **EIO_NODEV**).

Table 5-1 summarizes the software input channel assignments.

Table 5-1. PK2500 Software Input Channel Assignments

Protected Digital Input	Software Channel Assignment	Protected Digital Input	Software Channel Assignment
IN-00	0	IN-08	8
IN-01	1	IN-09	9
IN-02	2	IN-10	10
IN-03	3	IN-11	11
IN-04	4	IN-12	12
IN-05	5	IN-13	13
IN-06	6	IN-14	14
IN-07	7	IN-15	15

- `int eioBrdDO(unsigned chanNum, char state)`

Changes the state of an output channel.

PARAMETERS: `chanNum` must range from 0 (for OUT-00) through 11 (for OUT-11).

`state` is 0 if and only if the corresponding output is to be disabled “OFF,” or 1 if and only if the corresponding output is to be enabled “ON.”

RETURN VALUE:

- 0 if and only if `chanNum` is within range.
- -1 if and only if `chanNum` is out of range, that is, `chanNum` is greater than 11 (`eioErrorCode` is bit-ored with `EIO_NODEV`).

Table 5-2 summarizes the software output channel assignments.

Table 5-2. PK2500 Software Output Channel Assignments

High-Current Output	Software Channel Assignment	High-Current Output	Software Channel Assignment
OUT-00	0	OUT-06	6
OUT-01	1	OUT-07	7
OUT-02	2	OUT-08	8
OUT-03	3	OUT-09	9
OUT-04	4	OUT-10	10
OUT-05	5	OUT-11	11

The sample program `DIO1.C` turns the PK2500 into a relay. If digital input IN-00 (input channel 0) is grounded, the digital output OUT-00 (output channel 0) is disabled (OFF). Otherwise, the digital output is enabled.

DIO1.C

```
#use ezio.lib           // general I/O definitions
#use eziopk25.lib      // pk2500 specific defs
#define IN_CHAN 0      // define input channel
#define OUT_CHAN 0     // define output channel
main() {
    eioBrdInit(0);
    while (1) {        // do this indefinitely
        eioBrdDO(OUT_CHAN,eioBrdDI(IN_CHAN) );
        hitwd();      // hit watchdog
    }
}
```

Level-Sensitive Interrupts

The PK2500 can generate two level-sensitive processor interrupts under software control. The interrupts respond to a logic level “0” or OFF.



A logic level of 0 or OFF condition indicates that an input voltage is below the threshold of 2.5 V DC.

Protected digital inputs IN-06 and IN-07 are connected directly to the /INT0 and /INT1 interrupt lines of the Z180 processor, as shown in Figure 5-1.

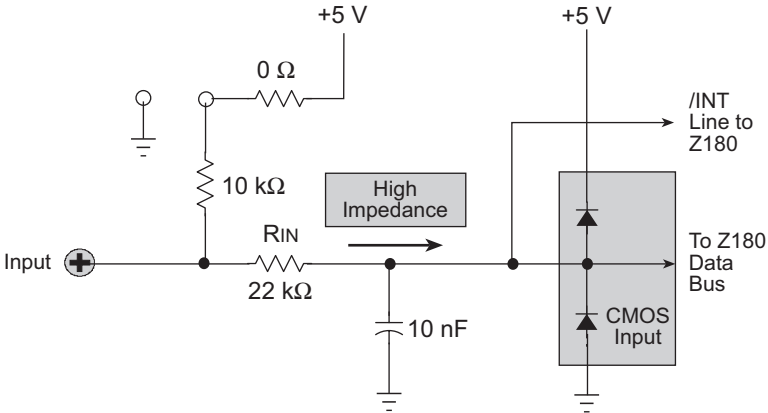


Figure 5-1. Level-Sensitive Inputs IN-06 and IN-07

The factory default is for the level-sensitive interrupts to be disabled (OFF), allowing inputs IN-06 and IN-07 to be used as standard protected digital inputs.

When level-sensitive interrupts are required, /INT0 and /INT1 can be enabled (ON) or disabled (OFF) by including the following commands in the application.

- **ISET (ITC , 0)**
Enables /INT0.
- **IRES (ITC , 0)**
Disables /INT0.
- **ISET (ITC , 1)**
Enables /INT1.
- **IRES (ITC , 1)**
Disables /INT1.

The sample program **EXTINT.C** in the Dynamic C **SAMPLES\PK25xx** subdirectory demonstrates how to set up protected digital inputs IN-06 and IN-07 to generate level-based interrupts, and demonstrates how the interrupts are captured by interrupt routines.

Interrupt Service Routines

The Z180 jumps to and executes interrupt service routines when it receives an interrupt request from /INT0 or /INT1.



Refer to the Dynamic C reference manuals for instructions on writing interrupt service routines (ISR).

Refer to the Zilog **Z80180/Z180 MPU User's Manual** for more information on using Z180 interrupts.



When /INT0 and /INT1 are enabled (ON), a logic level of 0 on IN-07 or IN-08 stops the Z180 processor so that it performs the ISR specified in the application. All other activities are stopped until the ISR is complete. Enabling /INT0 or /INT1 can severely affect the interrupt latency of other types of interrupts, such as those generated by programmable reload timers, the DMA, and UARTs.

Contact Z-World Technical Support at (530)757-3737 for assistance.



/INT0 and /INT1 can only be used when the PK2500's inputs are in the pulled-up configuration, which is the factory default. The PK2500 will malfunction if these interrupts are enabled with the inputs in a pulled-down configuration.

Pulse-Width Modulation Outputs

The following pulse-width modulation (PWM) functions are located in the **EZIOPK25.LIB** library.

- **void _eioSetupAO1st ()**
Initializes the PWM hardware.
`_eioSetupAO1st` must be called before using `eioBrdAO`.
- **int eioBrdAO(unsigned chanNum, unsigned state)**
Specifies the duty cycle for a particular output channel.
 - **chanNum** is a number ranging from 0 (for OUT-00) to 3 (for OUT-03).
 - **state** is a placeholder for a number ranging from 0 (to turn the channel off) to 256 (to turn the channel on, 100% duty cycle). The duty cycle is $state/256$ (for example, 128 for a 50% duty cycle, and 64 for a 25% duty cycle).



Although the PWM feature is available on up to six outputs in one bank, `eioBrdAO` only supports four PWM outputs. Refer to the more advanced PWM programming features later in this chapter if more than four PWM outputs are needed.

- **int _eioBrdAORf ()**
Refreshes the DMA counter and address pointer.
An application must call `_eioBrdAORf` every 25 ms (or more frequently) after `_eioSetupAO1st` is called.
The function returns -1 if the DMA count is zero (PWM has stopped), and it returns 0 otherwise. If the function returns -1, it means the driver is either not initialized (by calling `_eioSetupAO1st`), or `_eioBrdAORf` is called less frequently than every 25 ms.



While these PWM functions are simple to use, there are side effects associated with them. The functions use the Z180's built-in DMA hardware, which limits the communication speed of the Z180's Serial Port 1 to 4800 bps. The Z180 also runs 8% slower.

When using PWM functions, remember that the application must call `_eioBrdAORf` at least every 25 ms.

The sample program `AO1.C` ramps the PWM output from OUT-00 up and down.

```

#include ezio.lib           // general I/O definitions
#include eziopk25.lib      // pk2500 specific defs
#define OUT_CHAN 0        // define output channel
main() {
    auto unsigned dutyCycle;
    auto int sign;
    eioBrdInit(0);        // initialize general I/O
    _eioSetupAO1st();    // initialize PWM
    dutyCycle = 0;       // duty cycle starts at 0
    sign = 1;            // ramp up
    while (1) {          // do this forever
        eioBrdAO(OUT_CHAN,dutyCycle); // change cycle
        if (_eioBrdAORf() == -1) break; // refresh OK?
        if (dutyCycle == 256) sign = -1; // reverse
        else if (dutyCycle == 0) sign = 1; // reverse
        dutyCycle = dutyCycle + sign; // ramp
        hitwd();          // hit watchdog
    }
    printf("AO refresh failed\n");
}

```

References to Additional Software Features

- For **real-time clock** information, refer to descriptions of functions `tm_rd` and `tm_wr` in the Dynamic C reference manuals.
- For information on **communication ports**, refer to descriptions of the AASC libraries in the Dynamic C *Function Reference* manual. RS-485 driver switching is covered by `on_485` and `off_485` in the Dynamic C *Function Reference* manual.
- **Watchdog** information is provided with the `hitwd` in the Dynamic C *Function Reference* manual.
- **Simulated EEPROM** information is provided with the `ee_rd` and `ee_wr` in the Dynamic C *Function Reference* manual.
- **Power Fail Flag** information is provided with the `_sysIsPwrFail` and `sysIsPwrFail` functions in the Dynamic C *Technical Reference* manual.

Advanced Input/Output Programming

The previous section explains how to use Dynamic C functions to read inputs and write outputs. This section is intended for software engineers who need to optimize their code.



The topics discussed in this section assume an understanding on the part of the reader about the Z180 and its peripheral architecture. These details are contained in the Z180 technical manuals, available from Z-World. For more information, call your Z-World Sales Representative at (530)757-3737.

Table 5-3 provides a concise I/O map for the PK2500.

Table 5-3. PK2500 I/O Map

I/O Address	Description	Comments
0x4140 (write)	RS-485 Driver	Data bit 0 indicates on/off: 1 = on, 0 = off
0x4141 (write)	LED D1 (RUN)	Same as above
0x4142 (write)	LED D2 (USER)	Same as above
0x4143 (write)	Relay 0	Same as above
0x4144 (write)	Relay 1	Same as above
0x4145 (write)	OUT-00 to OUT-05	Data bits 0,1,2 select which high-current driver, bit 7 indicates on/off: 1 = on, 0 = off
0x4146 (write)	OUT-06 to OUT-11	Data bits 0,1,2 select which high-current driver, bit 7 indicates on/off: 1 = on, 0 = off
0x4147 (write)	A/D chip select	Data bit 0 indicates on/off: 1 = not selected, 0 = selected
0x4160 (read)	IN-00 to IN-07	Data bit x for state of IN-x
0x4161 (read)	IN-08 to IN-15	Data bit x for state of IN-x + 8
0x4142 (read)	A/D output data	Data bit 0 indicates current state of A/D serial output, address bit A00 simultaneously supplies a 0 to the A/D serial input
0x4143 (read)	A/D output data	Data bit 0 indicates current state of A/D serial output, address bit A00 simultaneously supplies a 1 to the A/D serial input
0x4144 (read)	A/D end of conversion	Data bit 0 indicates state of the A/D EOC flag: 0 = busy; 1 = done

Digital Input Addressing Details

Read I/O address **0x4160** for the states of inputs IN-00 to IN-07. In this I/O read, bit 0 corresponds to IN-00 and bit 7 corresponds to IN-07. Read I/O address **0x4161** for states of inputs IN-08 to IN-15. In this I/O read, bit 0 corresponds to IN-08, and bit 7 corresponds to IN-15. Table 5-4 lists the address structure of the digital inputs.

Table 5-4. PK2500 Digital Input States

0x4140							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
IN-07	IN-06	IN-05	IN-04	IN-03	IN-02	IN-01	IN-00
0x4141							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
IN-15	IN-14	IN-13	IN-12	IN-11	IN-10	IN-09	IN-08

Table 5-5 provides the Dynamic C commands to read each individual input.

Table 5-5. PK2500 Digital Input States

Channel	Function	Channel	Function
IN-00	IBIT (0x4140,0)	IN-08	IBIT (0x4141,0)
IN-01	IBIT (0x4140,1)	IN-09	IBIT (0x4141,1)
IN-02	IBIT (0x4140,2)	IN-10	IBIT (0x4141,2)
IN-03	IBIT (0x4140,3)	IN-11	IBIT (0x4141,3)
IN-04	IBIT (0x4140,4)	IN-12	IBIT (0x4141,4)
IN-05	IBIT (0x4140,5)	IN-13	IBIT (0x4141,5)
IN-06	IBIT (0x4140,6)	IN-14	IBIT (0x4141,6)
IN-07	IBIT (0x4140,7)	IN-15	IBIT (0x4141,7)

Digital Output and Relay Output Addressing Details

Write to I/O address **0x4140** to set the output states of OUT-00 through OUT-11. The lowest three bits is a number that specifies one of the 12 channels (0 for OUT-00). The most significant bit indicates whether the designated channel is to be enabled (ON) if the bit is a 1, or disabled (OFF) if the bit is a 0.

Table 5-6 summarizes the digital output states.

Table 5-6. PK2500 Digital Output States

I/O Address	Channel	ON	OFF
0x4140	RS485	xxxxxxx1	xxxxxxx0
0x4141	LED D1	xxxxxxx1	xxxxxxx0
0x4142	LED D2	xxxxxxx1	xxxxxxx0
0x4143	Relay 1	xxxxxxx1	xxxxxxx0
0x4144	Relay 2	xxxxxxx1	xxxxxxx0
0x4145	OUT-00	1xxxx000	0xxxx000
0x4145	OUT-01	1xxxx001	0xxxx001
0x4145	OUT-02	1xxxx010	0xxxx010
0x4145	OUT-03	1xxxx011	0xxxx011
0x4145	OUT-04	1xxxx100	0xxxx100
0x4145	OUT-05	1xxxx101	0xxxx101
0x4146	OUT-06	1xxxx000	0xxxx000
0x4146	OUT-07	1xxxx001	0xxxx001
0x4146	OUT-08	1xxxx010	0xxxx010
0x4146	OUT-09	1xxxx011	0xxxx011
0x4146	OUT-10	1xxxx100	0xxxx100
0x4146	OUT-11	1xxxx101	0xxxx101

For example, to turn OUT-04 on, the lowest three bits should be binary 100, and the most significant bit should be a 1, making the byte to write (binary) **1xxxx100** (xxxx means the values of these bits does not matter). **0x84** is one of the many variant representations of binary **1xxxx100**.

Analog-to-Digital Converter Addressing Details

Three I/O addresses are used to communicate with the A/D converter chip. The A/D active-low chip select input is controlled by I/O address **0x4147**. Write a 0 to turn on the chip select line, and write a 1 to turn it off, as shown below.

- **output (0x4147,0)**
Enables the A/D converter chip.
- **output (0x4147,1)**
Disables the A/D converter chip.

The A/D “End of Conversion” status line, EOC, is read using the data bit 0 of I/O address **0x4144**. A 1 is read when a conversion has been completed, and a 0 is read when the A/D converter is busy.

Converted A/D data are read a bit at a time using I/O address **0x4142** and **0x4143**. As each bit is read, the command for the next conversion is supplied to the A/D converter chip, also a bit at time, depending on which address is used. The least significant address bit, **A00**, does the trick: reading from **0x4142** sends a 0 to the A/D converter chip, while reading from **0x4143** sends a 1.

Table 5-7 summarizes the A/D converter input states.

Table 5-7. PK2500 A/D Converter States

0x4144							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	EOC
0x4142 (read data and write a 0)							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	A/D data
0x4143 (read data and write a 1)							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	A/D data



See “Analog-to-Digital Converter Inputs” in Chapter 4, “System Development,” for more information on the A/D converter inputs.

LED Addressing Details

LED D5 (Run) corresponds to I/O address **0x4141**; write 1 to turn the LED on, write 0 to turn the LED off.

- **outport(0x4141, 1)**
Enables LED D5 (Run) “ON.”
- **outport(0x4141, 0)**
Disables LED D5 (Run) “OFF.”

LED D4 corresponds to I/O address **0x4142**, write 1 to turn the LED on, write 0 to turn the LED off.

- **outport(0x4142, 1)**
Enables LED D4 (User) “ON.”
- **outport(0x4142, 0)**
Disables LED D4 (User) “OFF.”

RS-485 Driver Addressing Details

The RS-485 driver corresponds to the I/O address **0x4140**. Write 1 to enable the RS-485 driver, write 0 to disable the RS-485 driver.

- **outport(0x4140, 1)**
Enables RS-485 driver “ON.”
- **outport(0x4140, 0)**
Disables RS-485 driver “OFF.”

PWM Addressing Details

The pulse-width modulation (PWM) driver on the PK2500 is fairly complicated. This is because it uses the clock output from Communication Port 1 (**CKA1**) to drive the request line DMA Channel 0 in the edge-detection mode. The simple interface described previously (**eiOBrdAO**) provides PWM support for OUT-00 to OUT-05. If the application requires more PWM channels, specific frequencies, or precision, the application engineer may need to make trade-offs.

This section describes how PWM channels are driven, as well as how to customize PWM resource allocation to compromise the number of modulated channels, frequency and resolution.

Step 1. Determine the number of channels, frequency, resolution.

A pulse-width modulated waveform has a frequency and a resolution. The frequency states how many times the pattern repeats itself in a second (Hz). The resolution states how many divisions within one waveform can be resolved (distinguished). As a collection, the PWM driver also needs

to know the total number of channels to be pulse-width modulated. All channels are assumed to have the same frequency and resolution.

The clock output from Communication Port 1 (**CKA1**) must have a frequency $f_1 = N_{\text{ch}} \times f_w \times R_w$, where f_1 is the frequency of **CKA1**, N_{ch} is the number of channels PW modulated, f_w is the frequency of each channel, and R_w is the resolution in number of divisions per wave.

For example, the driver interface, `_eioSetupA01st`, makes the following assumptions: $N_{\text{ch}} = 4$, $f_1 = 76,800$ Hz, and $R_w = 256$. Consequently, $f_w = 76,800 \text{ Hz} / (4 \times 256) = 75$ Hz.

Step 2. Declare storage for the waveform pattern buffer (WPB).

Memory must be allocated to store the waveform pattern.

Step 3. Set up the waveform.

The PWM functions use the Z180's built-in DMA mechanism to transfer PWM "edges" from memory to the high-current ports at specific time intervals. Each edge is a byte whose least-significant three bits select one of the high-current outputs, OUT-00 through OUT-11. The most significant bit is a 1 to turn the specified port on (rising PWM "edge") or a 0 to turn the specified port off (falling PWM "edge"). Edges for the channels being pulse-width modulated are then grouped into composite transitions.

Each composite transition is a series of edges, each representing one possible transition for an individual channel. For example, if OUT-00 and OUT-01 are the only pulse-width modulated channels, a composite transition consists of two bytes, one to specify a possible transition for channel OUT-01, the other to specify a possible transition for channel OUT-02.

Assume the first byte in the composite transition corresponds to OUT-01, and the second byte corresponds to OUT-02.

The composite PWM waveform is a series of composite transitions (CTs) that specify the duty cycle of the pulse-width modulated channels. For example, if OUT-00 is to be at 0.375 duty cycle, and OUT-01 is to be at 0.75 duty cycle, both with a resolution of 8 divisions per cycle, a simple waveform would be as follows.

CT1: turn OUT-00 on, turn OUT-01 on.

CT2: do nothing.

CT3: do nothing.

CT4: turn OUT-00 off.

CT5: do nothing.

CT6: do nothing.

CT7: turn OUT-01 off.

CT8: do nothing.

go back to CT1.

Outputting the byte **0x80** turns OUT-00 on, **0x00** turns OUT-00 off, **0x81** turns OUT-01 on, and **0x01** turns OUT-01 off.

The byte **0x07** is a “no-op” and does nothing. This is because OUT-07 corresponds to I/O address **0x4146**, not **0x4160**. Consequently, the composite transitions (with no-ops) can be translated into the following byte sequence to be sent to I/O address **0x4160**.

CT1: **0x80, 0x81**
CT2: **0x07, 0x07**
CT3: **0x07, 0x07**
CT4: **0x00, 0x07**
CT5: **0x07, 0x07**
CT6: **0x07, 0x07**
CT7: **0x07, 0x01**
CT8: **0x07, 0x07**
go back to CT1

The equivalent byte stream (contents in the waveform pattern buffer) is a repeating pattern of the following.

0x80, 0x81, 0x07, 0x07, 0x07, 0x07, 0x00, 0x07,
0x07, 0x07, 0x07, 0x07, 0x07, 0x01, 0x07, 0x07

The driver library provides a function, **dmappwmSetBuf**, that allows the application engineer to modify the content of the waveform pattern buffer.



The function **dmappwmSetBuf** is discussed later in this chapter in the “PWM Advanced Programming Functions” section.

Step 4. Set up the clock.

The DMA device transfer from memory to I/O port address **0x4160** is driven by falling edges on signal /DREQ0. Since /DREQ0 is connected to **CKA1** (the clock output of communication channel 1), the communication speed of Communication Channel 1 determines how frequently the DMA device transfers memory to I/O. Each transfer corresponds to one edge in the previous section.

The driver includes a function, **dmappwmInit**, that sets up the frequency of **CKA1**. The function is described in the API section.

The PWM interface sets up **CKA1** to clock at 76,800 Hz in the call **_eioSetupAO1st()**.



The function **dmappwmInit** is discussed later in this chapter in the “PWM Advanced Programming Functions” section.

Refer to a Zilog or Hitachi user’s manual for the Z80180/Z180 or the 64180 to get a comprehensive explanation of how to set up the frequency of **CKA1**.

Step 5. Refresh the DMA counter and source address.

The DMA device does not automatically reload the counter and source address registers when the specified amount of bytes is transferred. When the DMA finishes transferring the specified amount of bytes, it stops and optionally causes an interrupt. In other words, the PWM waveform is abruptly ended when the DMA finishes.

To overcome this limitation, the application program must periodically “refresh” the counter and source address registers of the DMA. The refresh should check whether the counter is less than a critical number. If so, both the counter and the source address registers must be “rewound” to a previous state (a larger counter value and a corresponding lower source address).

Note that the PWM waveforms cannot be disrupted in the course of refreshing the registers. In other words, the previous state to which the refresh routine restores must be phase-synchronized with the PWM waveforms at the moment.

The driver library provides a refresh routine, `_eioBrdaORf`, to refresh the DMA counter and source address registers. `_eioBrdaORf()` can be called from a preemptive task or from the main program. The refresh routine must be called frequently enough so that the DMA counter never reaches 0. The following inequality states the requirement.

$$f_r \geq f_1 / (l_{\text{wpb}} / 2)$$

in which f_r is the refresh frequency, f_1 is the frequency of CKA1, and l_{wpb} is the total length of the waveform pattern buffer.

For example, `_eioSetupA01st()` sets up $f_1 = 76,800$ Hz, and $l_{\text{wpb}} = 4096$. As a result, the application engineer must ensure $f_r \geq 37.5$ Hz.

Step 6. Change duty cycles.

Once the PWM waveforms are up and running, the application may need to change the duty cycles for the channel(s). This poses two problems. First, the change should only be done to the channel that needs a change of duty cycle, all other channels should remain the same. Second, the change must become effectively phase-synchronized with the current waveform.

The solution to the first problem depends on how the edges are represented. In particular, it depends on whether no-operation (no-op) edges are used. If the no-op edges are used, changing duty cycle is a matter of moving the edges that are not “no-op.”

In the example in Step 3, change the waveform from

```
0x80, 0x81, 0x07, 0x07, 0x07, 0x07, 0x00, 0x07,  
0x07, 0x07, 0x07, 0x07, 0x07, 0x01, 0x07, 0x07
```

to

```
0x80, 0x81, 0x07, 0x07, 0x00, 0x07, 0x07, 0x07,  
0x07, 0x07, 0x07, 0x07, 0x07, 0x01, 0x07, 0x07
```

to change the duty cycle of OUT-00 to 0.25. The underlined edges are the only ones affected.

Of course, the waveform pattern buffer may have the pattern repeated many times. Each occurrence of the pattern in the buffer must be modified in the same way.

Even though the use of “no-op” edges seems to be compute-time inexpensive, it does require the application to maintain the location of the non-no-op edges. In other words, the application must maintain a duty cycle variable for each channel as well as a buffer for the waveform pattern.

Recall that the second problem of changing the duty cycle is the requirement for the change to be phase-synchronized to the current waveform. Many of the issues are similar to those of refreshing the DMA counter and pointer. The driver software library provides the function `dmapwmSwBuf` to switch waveform pattern buffers.



The sample programs in the Dynamic C `SAMPLES` subdirectory provide further detailed explanations.

PWM Advanced Programming Functions

- `void dmapwmSetBuf(char *pBufStart,
char bufLength256, unsigned step,
char outChar)`

Formats part of the waveform pattern buffer for DMA-driven PWM.

`pBufStart` points to the first byte to be formatted. Note that `pBufStart` does not always have to point to a 256-byte aligned address.

`bufLength256` is the length of the buffer, including the overflow area.

`step` is the number of bytes to skip between outputting `outChar`.

`outChar` is the actual bytes to send to the I/O address.

In other words, starting at the address pointed to by `pBufStart`, `dmapwmSetBuf` changes every `step` byte to `outChar` for `bufLength256` many 256-byte units.

- **void dmapwmSwBuf(unsigned newBuf256)**

Use two buffers to facilitate all-or-none duty cycle transitions. While one buffer is being used by the DMA to generate the PWM output, modify the other buffer for the new PWM pattern. When the new buffer is ready, this function should be called to switch to use the buffer at the address pointed to by **newBuf256** in 256-byte units.

- **char *dmapwmBufBeg(char *bufPtr)**

The buffer used by the PWM mechanism starts at 256-byte boundaries. Normal data definition declarations such as

```
char buffer[0x2000]
```

start at byte boundaries. **dmapwmBufBeg** returns a character pointer that points to the first 256-byte aligned root address larger than or equal to the parameter **bufPtr**.

- **void dmapwmInit(**
 unsigned phyBuffer256,
 unsigned bufSize256,
 unsigned resSize256,
 unsigned ioAddr,
 char cka1rate)

Initializes the DMA PWM mechanism.

When the function returns, **CKA1** of Communication Port 1 generates clock pulses at a **cka1rate** of approximately 19.2 kHz to /DREQ0. DMA Channel 0 would then transfer memory to I/O for each clock pulse falling edge.

- **phyBuffer256** is the 256-byte aligned physical address of the buffer in 256-byte units. In general, if the buffer is defined as an array in root memory (that is, it is of type **char ***), the following expression should be passed to this parameter.

```
(unsigned) ((xaddr(buffer)+255)>>8)
```

in which **buffer** is a pointer of type **char *** to the array.

- **bufSize256** is the size of the buffer, in 256-byte units. This size should not include the overflow area.
- **resSize256** is the size of the overflow area in 256-byte units.
- **ioAddr** is the port to which the DMA should transfer memory content.
- **cka1rate** is the clock rate generated by **CKA1** in multiples of 19.2 kHz. The allowed values are 2, 4, and 8.

Blank



APPENDIX A: TROUBLESHOOTING

This appendix provides procedures for troubleshooting system hardware and software. The following sections are included.

- Out of the Box
- Dynamic C Will Not Start
- Dynamic C Loses Serial Link
- PK2500 Repeatedly Resets
- Common Programming Errors

Out of the Box

Check the items mentioned in this section before starting development.

- Do not connect any RS-485 equipment or I/O devices until the PK2500 has been verified to run standalone.
- Verify that the entire host system has good, low-impedance, separate grounds for analog and digital signals. Often the PK2500 is connected between the host PC and another device. Any differences in ground potential from unit to unit can cause serious problems that are hard to diagnose.
- Do not connect analog ground to digital ground anywhere.
- Double-check the connecting ribbon cables to ensure that all wires go to the correct screw terminals on the PK2500.
- Verify that the host PC's COM port works by connecting a good serial device to the COM port. Remember that COM1/COM3 and COM2/COM4 share interrupts on a PC. User shells and mouse drivers, in particular, often interfere with proper COM port operation. For example, a mouse running on COM1 can preclude running Dynamic C on COM3.
- Use the Z-World power supply that comes with the developer's kit. If another power supply must be used, verify that it has enough capacity and filtering to support the PK2500.
- Use the Z-World cables that come with the developer's kit. The most common fault of user-made cables is failure to properly assert CTS at the RS-232 port of the PK2500. Without CTS being asserted, the PK2500's RS-232 port will not transmit. Assert CTS by either connecting the RTS signal of the PC's COM port or looping back the PK2500's RTS.
- The PK2500 Developer's Kit comes with an RJ-12 cable. This cable looks very much like a standard U.S. telephone cable, except it uses 6-pin RJ-12 connectors, not 4-pin RJ-11 connectors.



A regular U.S. telephone RJ-11 connector will plug in but *will not work*.

- Experiment with each peripheral device connected to the PK2500 to determine how it appears to the PK2500 when powered up, powered down, and/or when its connecting wiring is open or shorted.

Dynamic C Will Not Start

If Dynamic C will not start, an error message such as **Target Not Responding** or **Communication Error** appears on the Dynamic C screen. The following situations and their possible resolutions are usually behind these error messages.

- *Wrong Baud Rate* — In rare cases, the baud rate has to be changed when using the Serial Interface Board for development.
- *Wrong Communication Mode* — Both sides must be talking RS-232.
- *Wrong COMPort* — A PC generally has two serial ports, COM1 and COM2. Specify the one being used in the Dynamic C **Target Setup** menu. Use trial and error, if necessary.
- *Wrong Operating Mode* — Communication with Dynamic C will be lost if the PK2500's jumper is set for standalone operation. Reconfigure the board for programming mode.

If all else fails, connect the serial cable to the PK2500 after powerup. If the PC's RS-232 port supplies a large current (most commonly on portable and industrial PCs), some RS-232 level converter ICs go into a nondestructive latch-up. Connect the RS-232 cable after powerup to eliminate this problem.

Dynamic C Loses Serial Link

If the program disables interrupts for a period greater than 50 ms, Dynamic C will lose its serial link with the application program. Make sure that interrupts are not disabled for a period greater than 50 ms.

PK2500 Repeatedly Resets

The PK2500 resets every second if the watchdog timer is not “hit.” If a program does not “hit” the watchdog timer, then the program will have trouble running in standalone mode. To “hit” the watchdog, make a call to the Dynamic C library function `hitwd`.

Common Programming Errors

- Values for constants or variables out of range. Table A-1 lists acceptable ranges for variables and constants.

Table A-1. Ranges of Dynamic C Function Types

Type	Range
int	-32,768 (-2^{15}) to +32,767 ($2^{15} - 1$)
long int	-2,147,483,648 (-2^{31}) to +2147483647 ($2^{31} - 1$)
float	1.18×10^{-38} to 3.40×10^{38}
char	0 to 255

- Mismatched “types.” For example, the literal constant **3293** is of type **int** (16-bit integer). However, the literal constant **3293.0** is of type **float**. Although Dynamic C can handle some type mismatches, avoiding type mismatches is the best practice.
- Counting up from, or down to, one instead of zero. In software, ordinal series often begin or terminate with zero, not one.
- Confusing a function’s definition with an instance of its use in a listing.
- Not ending statements with semicolons.
- Not inserting commas as required in function parameter lists.
- Leaving out ASCII space character between characters forming a different legal—but unwanted—operator.
- Confusing similar-looking operators such as **&&** with **&**, **==** with **=**, and **//** with **/**.
- Inadvertently inserting ASCII nonprinting characters into a source-code file.
- The DMA-driven PWM function drives the output correctly for a while, then some channels remain ON, while others remain OFF. The most likely cause is **_eioBrdAORf** is not called frequently enough. Either increase the frequency with which this function is called, increase the size of the waveform platform buffer, or slow down the clock **CKA1**.



APPENDIX B: SPECIFICATIONS

This appendix provides comprehensive PK2500 physical, electronic and environmental specifications. The following sections are included.

- Electrical and Mechanical Specifications
- Factory Default Jumper Positions
- Protected Digital Inputs
- High-Current Drivers

Electrical and Mechanical Specifications

Figure B-1 illustrates external dimensions for the PK2500.

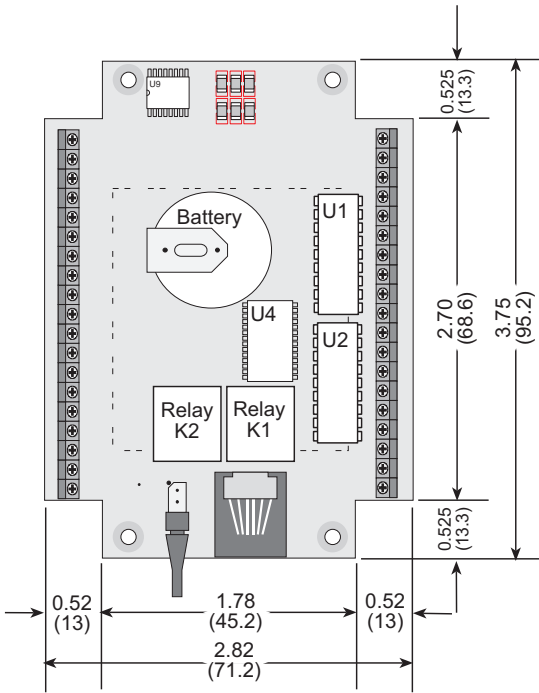


Figure B-1. PK2500 External Dimensions

Table B-1 lists electrical, mechanical, and environmental specifications for the PK2500.

Table B-1. PK2500 Specifications

Parameter	Specification
Board Size	2.82"W × 3.75"L × 1.18"H (71.2 mm × 95.2 mm × 30 mm)
Enclosure Size	2.96"W × 5.00"L × 1.83"H (75 mm × 127 mm × 46 mm)
Operating Temperature	-40°C to 70°C
Humidity	5% to 95%, noncondensing
Power Supply	9 V to 36 V DC, typ 75 mA at 25°C with 24 V switching power supply

continued...

Table B-1. PK2500 Specifications (concluded)

Parameter	Specification
Digital Inputs	<ul style="list-style-type: none"> • 10 protected digital inputs, -20 V to +24 V DC <ul style="list-style-type: none"> - 2 may be configured with level-sensitive interrupts - 2 may be configured as an RS-485 serial port • up to 6 digital outputs may be configured as protected digital inputs in groups of 3
Digital Outputs	<ul style="list-style-type: none"> • 12 high-current sinking, 75 mA/channel at 48 V (all channels ON) at 60°C • (optional) 6 or 12 high-current sourcing, 75 mA/channel at 30 V at 60°C • up to 6 of the digital outputs may be configured as protected digital inputs in groups of 3
A/D Converter Inputs	4 analog channels with 12-bit resolution, input voltage 0 V to 10 V; other ranges available as factory options
Relay Outputs	2 normally open with indicator LEDs, 60 W, 2 A (resistive) at 30 V DC, 500 mA (resistive) at 125 V AC, maximum switching voltage 125 V with arc suppression circuitry*
LEDs	4: 2 for relay outputs, 2 under software control
Processor	Z180
Clock	18.432 MHz
SRAM	128K standard
EEPROM	Flash memory may be used as EEPROM
Flash EPROM	128K standard
Serial Ports	<ul style="list-style-type: none"> • 2 RS-232 or 1 RS-232 with RTS/CTS and 1 RS-485 • programming is either via one RS-232 port or via Serial Interface Board 2
Serial Rate	Up to 57,600 bps
Watchdog/Supervisor	Yes
Time/Date Clock	Yes
Backup Battery	BR2325-1HG or equivalent, 3-year shelf life (power off), 10-year life in use (power on)
Connectors	Quick-disconnect screw terminal blocks, 3.5 mm pitch

* For CE compliance, the maximum relay switching voltage is less than 50 V AC or 75 V DC.

Factory Default Jumper Positions

Figure B-2 illustrates the header locations on the PK2500 main board. The jumpers configurations for these headers are listed in Table B-2.

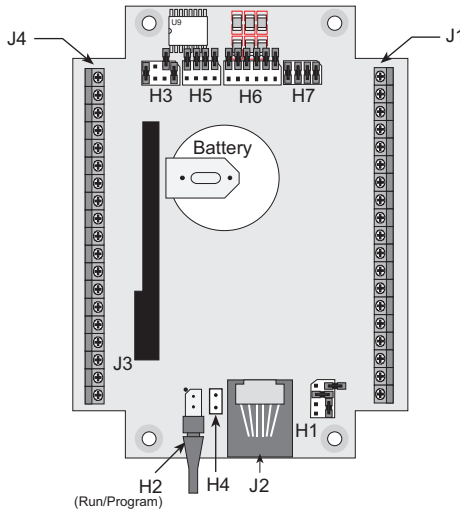


Figure B-2. Locations of PK2500 Headers

Table B-2. PK2500 Header Connections

Header	Pins	Description	Factory Default
H1	1-2	Connect to reset PK2500	Not connected
	3-5 7-8	Configure RS-485 communication for J1 pins 17 and 18	Pins 3-4 and 6-8 connected to enable protected digital inputs IN-08 and IN-09
	3-4 6-8	Configure protected digital inputs IN-08 and IN-09 for J1 pins 17 and 18	
H2	1-2	Connect pins 1-2 to enable program mode	Pins 1-2 connected
H3	4-6	RS-485 serial port on Z1	Pins 1-2 and 7-8 connected
	1-3 5-7	5-wire RS-232 serial port on Z0	
	1-2 7-8	Two 3-wire RS-232 serial ports (no handshaking)	

continued...

Table B-2. PK2500 Header Connections (concluded)

Header	Pins	Description	Factory Default
H5	1–2 3–4 5–6	Connect to configure J4 pin 4 as A/D voltage reference output	Not connected; J4 pin 4 is A/D converter input
	7–8	Connect to enable RS-485 termination resistor	Not connected
H6	1–2	Connect to configure J1 pin 7 as IN-15 (OUT-03 when disconnected)	Not connected
	3–4	Connect to configure J1 pin 8 as IN-14 (OUT-04 when disconnected)	Not connected
	5–6	Connect to configure J1 pin 9 as IN-13 (OUT-05 when disconnected)	Not connected
	7–8	Connect to configure J1 pin 14 as IN-12 (OUT-09 when disconnected)	Not connected
	9–10	Connect to configure J1 pin 15 as IN-11 (OUT-10 when disconnected)	Not connected
	11–12	Connect to configure J1 pin 16 as IN-10 (OUT-11 when disconnected)	Not connected
H7	1–2 3–4	Connect to configure OUT-00 to OUT-05 as sinking	Pins 1–2 and 3–4 connected to enable sinking outputs for OUT-00 to OUT-05
	1–3 2–4	Connect to configure OUT-00 to OUT-05 as sourcing	
	5–6 7–8	Connect to configure OUT-06 to OUT-11 as sinking	Pins 5–6 and 7–8 connected to enable sinking outputs for OUT-06 to OUT-11
	5–7 6–8	Connect to configure OUT-06 to OUT-11 as sourcing	



See Chapter 3, “Input/Output Configuration,” for more details on the jumper configurations.



While power must normally be disconnected when reconfiguring jumpers, power may be left on when connecting/disconnecting pins 1–2 on H1 to reset the PK2500.

Protected Digital Inputs

Table B-3 lists the specifications for the protected digital inputs.

Table B-3. Protected Digital Input Specifications

Protected Digital Inputs	Maximum Rating
Digital Input Voltage	-20 V to +24 V
Digital Input Current	15 mA
Frequency Response (worst case)	Faster than 656 Hz, not longer than 1.52 ms (input 5 V DC)

Frequency Response for IN-00 to IN-05, and IN-08 to IN-15

The protection network consists of a low-pass filter with a -3 dB point of 723 Hz. For example, if the driving source of a protected input is a step function, that step becomes available 1.38 ms later as a valid +5 V DC CMOS input to the PK2500's data bus.

The following formula shows how R_{IN} and C affect the frequency response of protected inputs IN-00 through IN-05 and IN-08 through IN-15.

$$f_c = [2\pi R_{IN} C]^{-1} = [(2\pi)(22 \times 10^3)(10 \times 10^{-9})]^{-1} = 723 \text{ Hz}$$

$$t = [f_c]^{-1} = 1.38 \text{ ms (at 0.707 of full input value)}$$

Figure B-3 shows the circuit for protected inputs IN-00 through IN-05, and IN-08 through IN-15 in the factory-default pulled-up configuration.

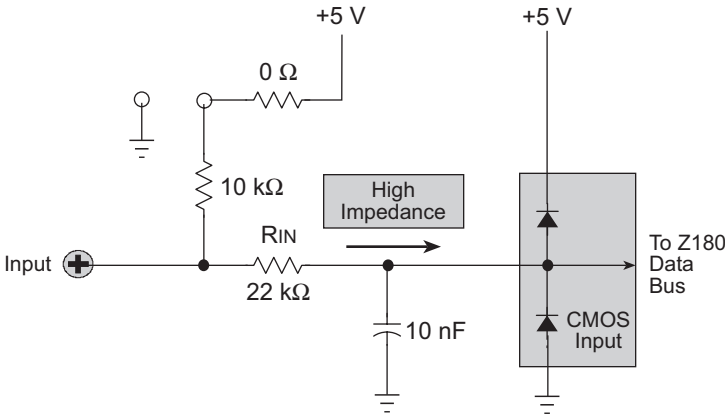


Figure B-3. Protected Digital Inputs IN-00 to IN-05, and IN-08 to IN-15

Frequency Response for IN-06 and IN-07

These two interrupt-ready inputs also have a protection network that consists of a low-pass filter. The -3 dB point is at 7.23 kHz because the input resistors have a lower resistance. For example, if the driving source of these two protected inputs is a step function, then that step voltage registers approximately 138 μ s later as a valid +5 V DC CMOS input to the PK2500's data bus.

Figure B-4 shows the circuit for IN-06 and IN-07 in the factory-default pulled-up configuration.

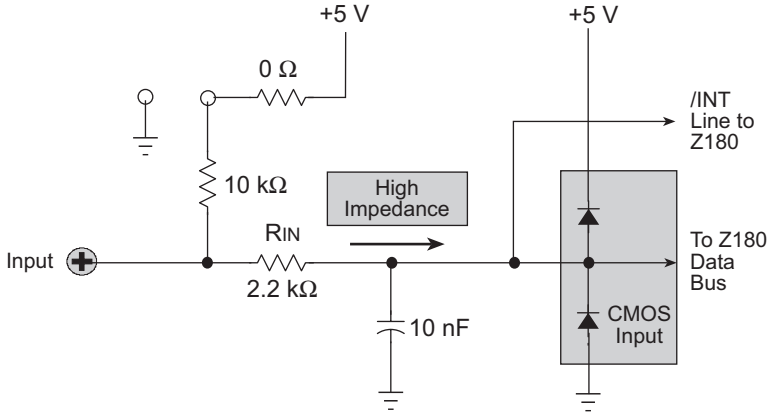


Figure B-4. Level-Sensitive Inputs IN-06 and IN-07

Customization

Frequency Response and Input Range

A faster frequency response is possible by replacing R_{IN} with a smaller resistor. For example, if the digital input is being driven by a +5 V DC CMOS-compatible driver, R_{IN} may be replaced with a 0 Ω valued 0805 resistor.



A 0 Ω resistor for R_{IN} will adversely affect the PK2500's noise immunity.



The PK2500 may be ordered with a customer-specified resistor installed by the factory at R_{IN}. For more details, contact your Z-World Sales Representative at (530)757-3737.

Default Pull-Up Assignments

All protected digital inputs are factory-configured with +5 V DC pull-up resistors. Since there are two banks (or groups) of inputs, each bank can be uniquely pulled up to +5 V DC or pulled down to ground by two separate permanent surface mount 0 Ω resistors.

- Bank **ONE** includes the dedicated protected digital inputs IN-00 to IN-03. Bank **ONE** is assigned via JP15 (0 Ω resistor).
- Bank **TWO** includes all the other inputs (IN-04 through IN-15), and is configured with JP14 (0 Ω resistor).

High-Current Drivers

Table B-4 lists the high-current driver characteristics when sinking drivers or sourcing drivers are used.

Table B-4. High-Current Driver Characteristics


Characteristic	Sinking Driver 	Sourcing Driver
IC Model Number	ULN2803A	UDN2985A
Channels	6 available	6 available
Max. Current per Channel (all channels ON)	75 mA @ 60°C 125 mA @ 50°C	75 mA @ 60°C 125 mA @ 50°C
Voltage Source Range	2 V to 48 V DC	15 V to 30 V DC
Package Power Dissipation	2.2 W (1.2 W @ 60°C)	2.2 W (1.2 W @ 60°C)
Max. Current (all channels ON)	1.38 A	1.38 A
Max. Collector-Emitter Voltage (V_{CE})	1.6 V	1.6 V
Derating	18 mW/°C (55°C/W)	18 mW/°C (55°C/W)
Output Flyback Diode (K)	Yes	Yes
Max. Diode-Drop Voltage (K)	2 V DC	2 V DC

Table B-5 lists safe operating conditions for the PK2500 high-current drivers.

Table B-5. High-Current Driver Safe Operating Range at 60°C

No. Outputs ON	Current/Channel	Percent Duty Cycle
6	125 mA	100
4	186 mA	100
2	375 mA	100
6	250 mA	50
4	375 mA	50
2	500 mA	50



Contact Z-World Technical Support at (530)757-3737 for further information about maximum operating conditions for the PK2500 high-current drivers.

“KA” and “KB”

The PK2500 has two separate common supply pins labeled KA and KB. KA is used for high-current outputs OUT-00 to OUT-05, and KB is used for OUT-06 to OUT-11. This allows different voltages to be used for each set of outputs.

The KA and KB connections perform two functions to the high-current driver circuits on the PK2500.

1. KA and KB supply power to the driver circuits inside the driver chips.
2. KA and KB also allow a diode internal to the driver chips to “snub” voltage transients produced during inductive kick (associated with switching inductive loads). Relays, solenoids, and speakers are examples of inductive loads.

The anodes of the protection diodes are common for each bank of channels, and so only one supply voltage can be used for each bank of high-current driver loads.

Figure B-5 and Figure B-6 illustrate the use of the K connections for sinking and sourcing configurations.

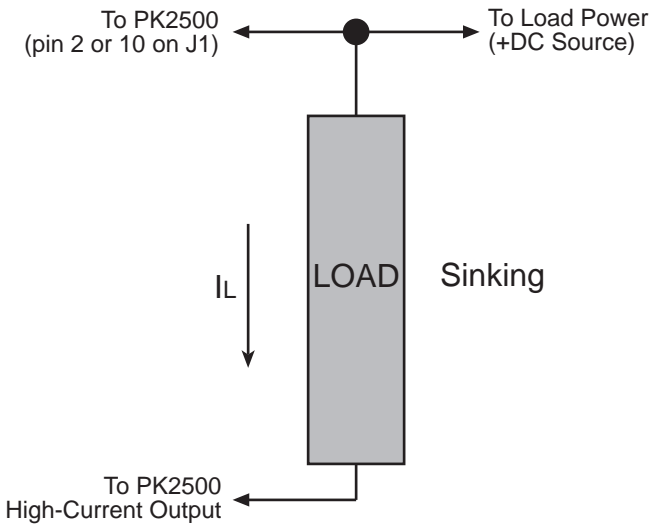


Figure B-5. PK2500 K Connection (Sinking Configuration)

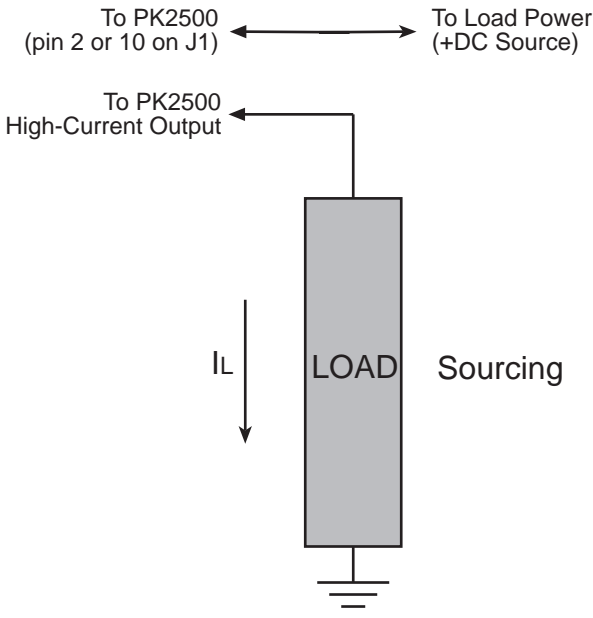


Figure B-6. PK2500 K Connection (Sourcing Configuration)



APPENDIX C: POWER MANAGEMENT

Appendix C provides detailed information about the power systems and how the PK2500 handles power failures.

Power-Failure Detection Circuitry

Figure C-1 shows the PK2500's power-failure detection circuit. Note that the 691 trips at 1.3 V DC.

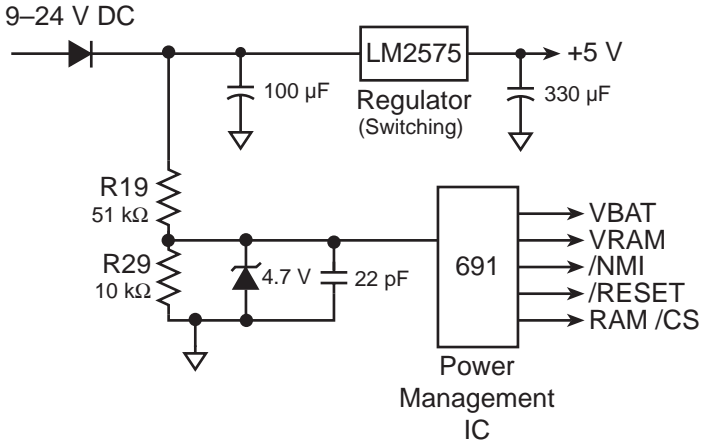


Figure C-1. PK2500 Power-failure Detection Circuitry

Power Failure Sequence of Events

Figure C-2 summarizes the events that occur as the input power fails.

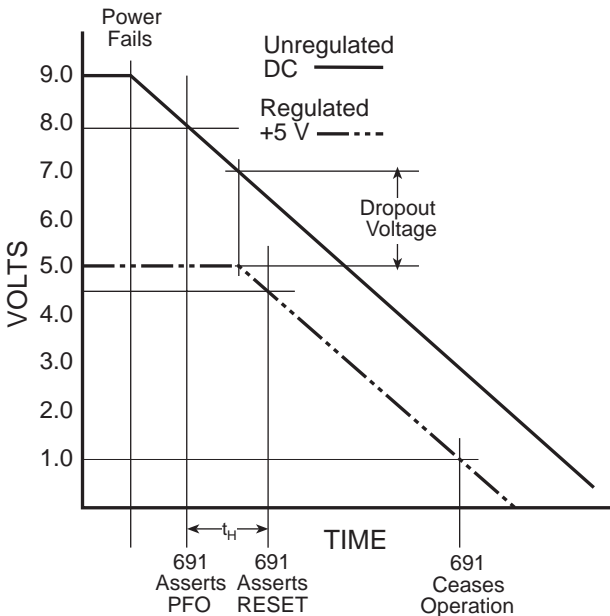


Figure C-2. Power Failure Sequence

First, the 691 power-management IC triggers a power-failure **/NMI** (non-maskable interrupt) when the *unregulated* DC input voltage (9 V to 24 V DC) falls in the range $8.02\text{ V} \leq +\text{DC} \leq 8.63\text{ V}$ (as determined by the voltage divider).

At some point, the raw input voltage level will not exceed the required regulated voltage level by the regulator's *dropout voltage* (see Figure C-2). The regulated output voltage then begins to droop. The 691 triggers a system reset when the *regulated* +5 V supply is in the range from 4.50 V to 4.75 V DC, allowing the power-failure routine a “holdup” interval, t_{H} , to store important state data. The 691 forces the chip select (**/CS**) of the SRAM to high (standby mode).

Tip

To increase the “holdup” interval t_{H} , use a power supply with a large capacitance. This will provide additional time for the PK2500 to execute a safe shutdown.

The time/date clock and SRAM switch to the lithium backup battery when the regulated voltage falls below the battery voltage of approximately 3 V.

The 691 keeps the reset (**/RESET**) enabled until the +5 V regulated voltage drops below 1 V. The 691 ceases operating at this point. By this time, the portion of the circuitry not battery-backed should have long since ceased functioning.

The ratio of the power supply's output capacitance to a circuit's current draw determines the actual duration of the holdup time interval, t_{H} .

A situation similar to a continuous low input (“brownout”) can occur if the power supply is overloaded. For example, when a high-current device such as a relay turns ON, the raw voltage supplied to the PK2500 may dip below 7.9 V. The interrupt routine does a shutdown. This shutdown turns the LED off, clearing the problem. However, if the cause of the overload persists, the system oscillates, alternately experiencing an overload and then resetting. To correct this situation, get a larger, “stiffer” power supply.

If the power cable is removed abruptly from the PK2500, then only the capacitors on the board are available, reducing computing time to a few microseconds. These times can vary considerably, depending on system's configuration and loads on the +5 V regulated and the 9 V to 24 V unregulated supplies.

The interval between power-failure detection and the entry to the power-failure interrupt routine is approximately 100 μs , or less if the Dynamic C **/NMI** communication is not in use.

Blank



APPENDIX D:
SINKING VS. SOURCING DRIVERS

Appendix D provides detailed information about sinking and sourcing high-current drivers.

The PK2500 has up to 12 high-current driver outputs that can be configured in groups of six as “sourcing” or “sinking.” The configuration is selected by installing the appropriate driver IC and by setting header H7. The factory-installed driver chip and default jumper settings are for “sinking” control (ULN2803).

Figure D-1 shows the locations of the driver ICs at U1 and U2. Figure D-2 shows the jumper configurations on header H7.

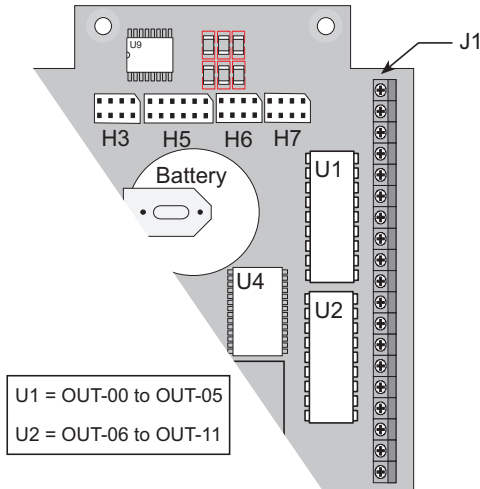


Figure D-1. PK2500 High-Current Output Driver Chips at U1 and U2

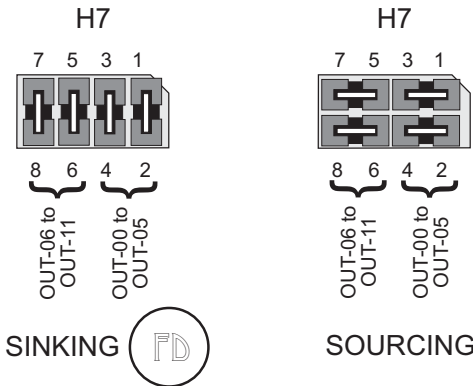


Figure D-2. PK2500 Header H7 Jumper Configurations



Note that pins 1–2/3–4 and 5–6/7–8 on header H7 may be set independently of each other to enable some outputs as sinking and the others as sourcing.

Selecting Sourcing or Sinking Drivers

The UDN2985 sourcing driver is included in the Developer's Kit. Since the driver ICs are socketed, the factory-default ULN2803 sinking drivers can be easily substituted.



The CM7200 Core Module needs to be removed to access the high-current output drivers. The CM7200 Core Module is plugged onto header J3 and held in place by one screw.



The PK2500 is available with one or both sourcing drivers factory-installed. For more information, call your Z-World Sales Representative at (530) 757-3737.

Sinking Driver (Low-Side Drive)

The ULN2803 sinking driver can handle up to 500 mA for any channel, or an absolute maximum of 0.75 A at 60°C, which represents an average of 125 mA per channel with all channels ON at 60°C. The absolute maximum power that the ULN2803 can dissipate is 2.2 W at 25°C (1.2 W at 60°C). The saturation voltage is a maximum of 1.6 V DC per channel. The sinking driver's source voltage must range from 2 V to 48 V.

Figure D-3 illustrates the sinking driver configuration.

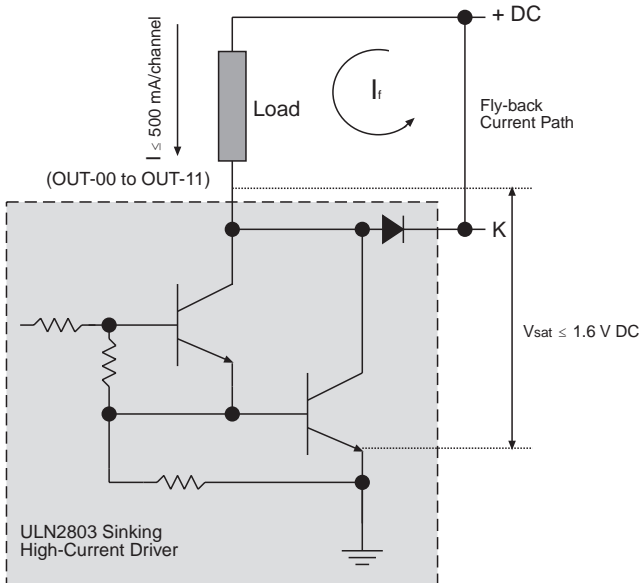


Figure D-3. Partial Schematic ULN2803 Sinking Driver

Sourcing Driver (High-Side Drive)

The UDN2985 sourcing driver can handle up to 500 mA for any channel, or an absolute maximum of 0.75 A at 60°C, which represents an average of 125 mA per channel with all channels ON at 60°C. The absolute maximum power that the UDN2985 can dissipate is 2.2 W at 25°C (1.2 W at 60°C). The saturation voltage is a maximum of 1.6 V DC per channel.

Figure D-4 illustrates the sourcing driver configuration.

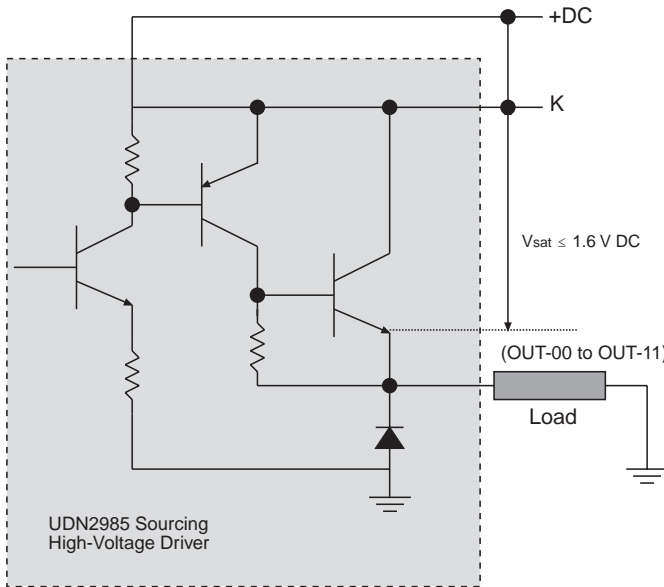


Figure D-4. Partial Schematic UDN2985 Sourcing Driver



The sourcing driver's source voltage must range from 3 V to 30 V DC. The minimum output sustaining voltage is 15 V DC. Operating the driver at more than 15 V without providing for energy dissipation may destroy the driver when an inductive load is connected.



See the Motorola (DL128) or Allegro (AMS 502Z) applications and technical data books for more information, including complete schematics, on sinking and sourcing high-current drivers.



APPENDIX E: **SERIAL INTERFACE BOARD 2**

Appendix E provides technical details and baud rate configuration data for Z-World's SIB2. The following sections are included.

- Introduction
- External Dimensions

Introduction

The SIB2 is an optional interface adapter used to program the PK2500. The SIB2 is contained in an ABS plastic enclosure, making it rugged and reliable. The SIB2 enables the PK2500 to communicate with Dynamic C via the Z180's clocked serial I/O (CSI/O) port, freeing both PK2500 serial ports for use by the application during programming and debugging.

The SIB2's 8-pin cable plugs into header JP1 of the PK2500's CM7200 core module, and a 6-conductor RJ-12 phone cable connects the SIB to the host PC. The SIB2 automatically selects its baud rate to match the communication rates established by the host PC (9600 bps, 19,200 bps, or 57,600 bps). However, the SIB2 determines the host's communication baud rate only on the first communication after reset. To change baud rates, change the COM baud rate, reset the target PK2500 (which also resets the SIB2), then select **Reset Target** from Dynamic C.

The SIB2 receives power and resets from the target PK2500 via the 8-pin connector. Therefore, do not unplug the SIB2 from the target PK2500 while power is applied. To do so could damage both the PK2500 and the SIB2; additionally, the target may reset.

The SIB2 consumes approximately 60 mA from the +5 V supply. Target-system current consumption therefore increases by this amount while the SIB2 is connected to the PK2500.



Never connect or disconnect the SIB2 with power applied to the controller.

External Dimensions

Figure E-1 illustrates the external dimensions for the SIB2.

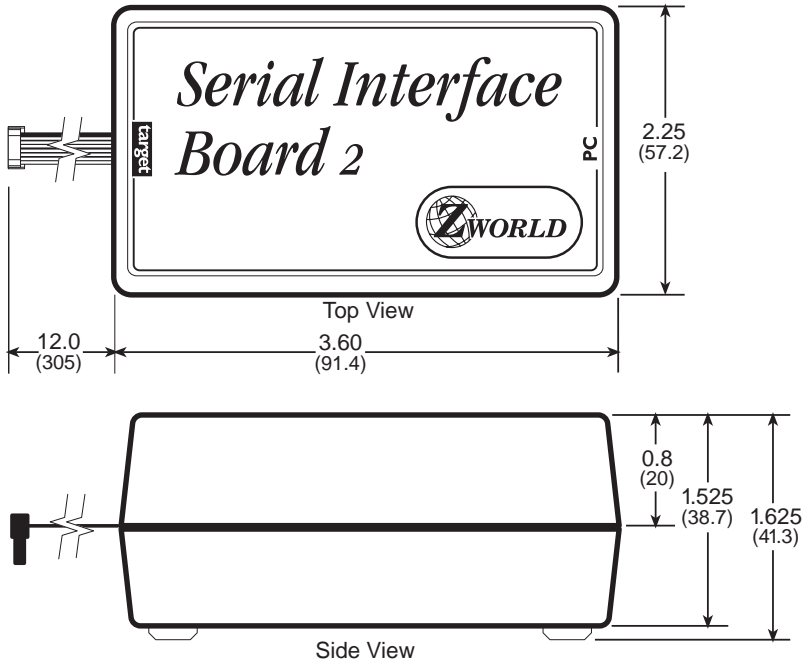


Figure E-1. SIB2 External Dimensions

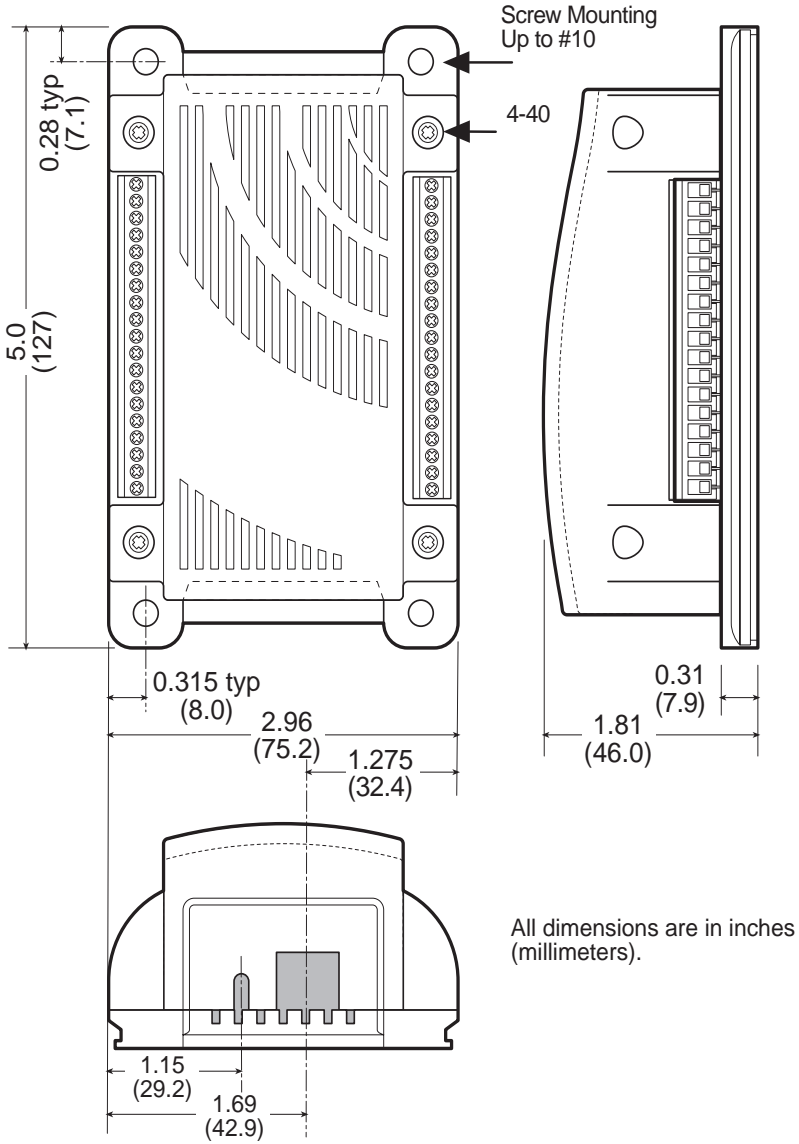
Blank



APPENDIX F: ENCLOSURE MOUNTING

Appendix F provides technical details for the optional PK2500 enclosure, and includes mounting suggestions for the enclosure.

Figure F-1 summarizes the dimensions of the optional enclosure.



All dimensions are in inches (millimeters).

Figure F-1. PK2500 Optional Enclosure Dimensions

The enclosure orientation affects the ability of the PK2500 to dissipate or remove heat from its circuitry to the outside ambient environment. Heat removal is crucial to reliable operation.

The preferred orientations of the PK2500 enclosure are shown in Figures F-2 to F-5 in order of recommended (1) to least desirable (5).

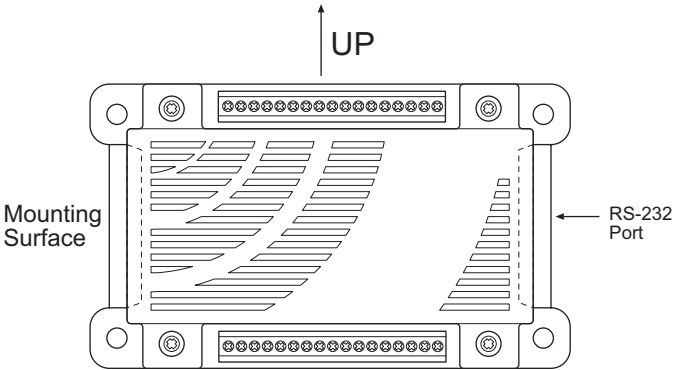


Figure F-2. Preferred PK2500 Enclosure Mounting

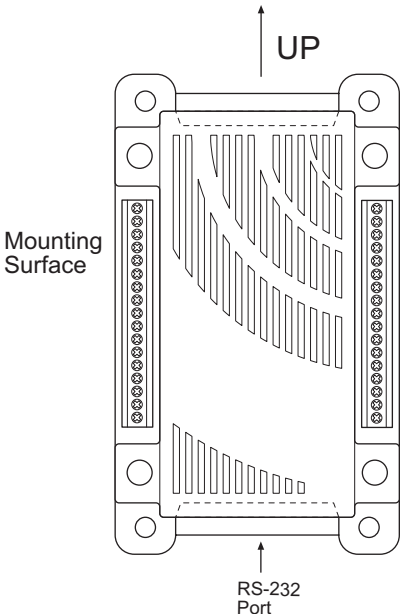


Figure F-3. Acceptable PK2500 Enclosure Mounting

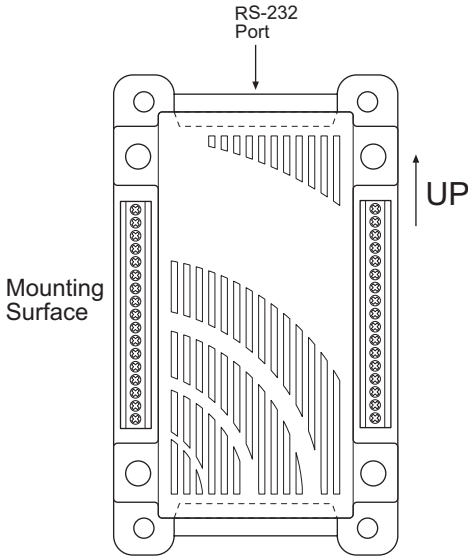


Figure F-4. Acceptable PK2500 Enclosure Mounting

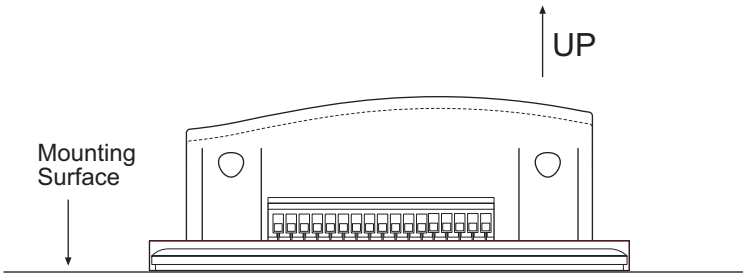
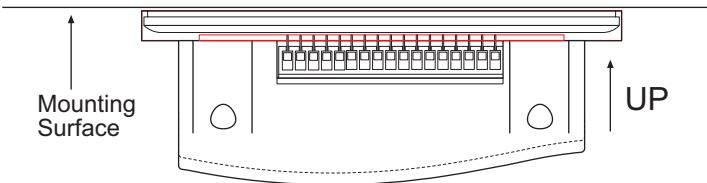


Figure F-5. Acceptable PK2500 Enclosure Mounting



**Figure F-6. PK2500 Enclosure Mounting
(not recommended)**



The mounting configuration shown in Figure F-6 is not recommended. Components on the PK2500 may overheat, leading to failure or reduced operating life.



*APPENDIX G: **NONVOLATILE STORAGE***

Appendix G provides information about the flash EPROM memory used in the PK2500.

Flash memory acts as EEPROM. Unlike many older Z-World controllers, the PK2500 does not have an EEPROM for nonvolatile storage of important system parameters. However, the PK2500 flash EPROM simulates EEPROM, and uses the same Dynamic C function calls as other Z-World controllers to read and write nonvolatile data.

The flash EPROM constants listed in Table G-1 apply to the PK2500.

Table G-1. PK2500 Flash EPROM Constants

Address	Definition
0	Startup mode. If 1, enter program mode. If 8, execute loaded program at startup.
1	Programming baud rate in multiples of 1200 bps. The factory value is 16 (19,200 bps).



APPENDIX H: ***I/O MAP AND INTERRUPT VECTORS***

Appendix H provides information on the PK2500's memory mapping, which is based on the CM7200 core module.

PK2500 Input/Output Map

The internal registers for the input/output devices built into the Z180 processor occupy the first 40 (hex) addresses of the I/O space. Table H-1 lists the addresses of these internal registers.

Table H-1. CM7200 Z180 Internal I/O Registers

Address	Name	Description
0x00	CNTLA0	Control Register A, Serial Channel 0
0x01	CNTLA1	Control Register A, Serial Channel 1
0x02	CNTLB0	Control Register B, Serial Channel 0
0x03	CNTLB1	Control Register B, Serial Channel 1
0x04	STAT0	Status Register, Serial Channel 0
0x05	STAT1	Status Register, Serial Channel 1
0x06	TDR0	Transmit Data Register, Serial Channel 0
0x07	TDR1	Transmit Data Register, Serial Channel 1
0x08	RDR0	Receive Data Register, Serial Channel 0
0x09	RDR1	Receive Data Register, Serial Channel 1
0x0A	CNTR	Clocked Serial Control Register
0x0B	TRDR	Clocked Serial Data Register
0x0C	TMDR0L	Timer Data Register, Channel 0, low
0x0D	TMDR0H	Timer Data Register, Channel 0, high
0x0E	RLDR0L	Timer Reload Register, Channel 0, low
0x0F	RLDR0H	Timer Reload Register, Channel 0, high
0x10	TCR	Timer Control Register
0x11–13	—	<i>Reserved</i>
0x14	TMDR1L	Timer Data Register, Channel 1, low
0x15	TMDR1H	Timer Data Register, Channel 1, high
0x16	RLDR1L	Timer Reload Register, Channel 1, low
0x17	RLDR1H	Timer Reload Register, Channel 1, high
0x18	FRC	Free-Running Counter
0x19–1F	—	<i>Reserved</i>
0x20	SAR0L	DMA Source Address, Channel 0, low
0x21	SAR0H	DMA Source Address, Channel 0, high
0x22	SAR0B	DMA Source Address, Channel 0, extra bits

continued...

Table H-1. CM7200 Z180 Internal I/O Registers (concluded)

Address	Name	Description
0x23	DAR0L	DMA Destination Address Channel 0, low
0x24	DAR0H	DMA Destination Address Channel 0, high
0x25	DAR0B	DMA Destination Address Channel 0, extra bits
0x26	BCR0L	DMA Byte Count Register, Channel 0, low
0x27	BCR0H	DMA Byte Count Register, Channel 0, high
0x28	MAR1L	DMA Memory Address Register, Channel 1, low
0x29	MAR1H	DMA Memory Address Register, Channel 1, high
0x2A	MAR1B	DMA Memory Address Register, Channel 1, extra bits
0x2B	IAR1L	DMA I/O Address Register, Channel 1, low
0x2C	IAR1H	DMA I/O Address Register, Channel 1, high
0x2D	—	<i>Reserved</i>
0x2E	BCR1L	DMA Byte Count Register, Channel 1, low
0x2F	BCR1H	DMA Byte Count Register, Channel 1, high
0x30	DSTAT	DMA Status Register
0x31	DMODE	DMA Mode Register
0x32	DCNTL	DMA/WAIT Control Register
0x33	IL	Interrupt Vector Low Register
0x34	ITC	Interrupt/Trap Control Register
0x35	—	<i>Reserved</i>
0x36	RCR	Refresh Control Register
0x37	—	<i>Reserved</i>
0x38	CBR	MMU Common Base Register
0x39	BBR	MMU Bank Base Reg
0x3A	CBAR	MMU Common/Bank Area Register
0x3B–3D	—	<i>Reserved</i>
0x3E	OMCR	Operation Mode Control Register
0x3F	ICR	I/O Control Register

Real-Time Clock Registers

Table H-2 provides the real time-clock IC's internal registers.

Table H-2. Real-Time Clock Internal Registers

Address	Data Bits	Symbol	Meaning	Range
0x4180	D0–D7	SEC1	seconds	0–9
0x4181	D0–D7	SEC10	10 seconds	0–5
0x4182	D0–D7	MIN1	minutes	0–9
0x4183	D0–D7	MIN10	10 minutes	0–5
0x4184	D0–D7	HOUR1	hours	0–9
0x4185	D0–D7	HOUR10	10 hours	0–2
0x4186	D0–D7	DAY1	days	0–9
0x4187	D0–D7	DAY10	10 days	0–3
0x4188	D0–D7	MON1	months	0–9
0x4189	D0–D7	MON10	10 months	0–1
0x418A	D0–D7	YEAR1	years	0–9
0x418B	D0–D7	YEAR10	10 years	0–9
0x418C	D0–D7	WEEK	weekdays	0–6
0x418D	D0–D7	TREGD	Register D	—
0x418E	D0–D7	TREGE	Register E	—
0x418F	D0–D7	TREGF	Register F	—

Other Input/Output Addresses

The I/O addresses listed in Table H-3 control the I/O devices that are external to the Z180 processor.

Table H-3. I/O Addresses for Devices External to Z180

Address	Description	Function
0x4140	RS-485 driver	Bit 0 indicates on/off: 1 to turn on, 0 to turn off
0x4141	LED D1 (RUN LED)	As above
0x4142	LED D2 (USER LED)	As above
0x4143	Relay 0	As above
0x4144	Relay 1	As above
0x4145	HVA0 to HVA5	Bits 0, 1, 2 indicate which high-current driver, bit 7 indicates on/off: 1 to turn on, 0 to turn off
0x4146	HVB0 to HVB5	Bits 0, 1, 2 indicate which high-current driver, bit 7 indicates on/off: 1 to turn on, 0 to turn off
0x4147	/ADCS	Bit 0 indicates on/off: 1 to turn on, 0 to turn off
0x4140 (read)	IN-00 to IN-07	Bit x for state of Inx
0x4141 (read)	OUT-01 to OUT-07	Bit x for state of Inx
0x4142 (read)	ADDIN	Bit 0 indicates on/off: 1 to turn on, 0 to turn off
0x4143 (read)	ADDIN	As above
0x4144 (read)	ADDEOC	As above

Interrupt Vectors

Table H-4 presents a suggested interrupt vector map. Most of these interrupt vectors can be altered under program control. The addresses are given here in hexadecimal, relative to the start of the interrupt vector page, as determined by the contents of the I-register. These are the default interrupt vectors set by the boot code in the Dynamic C EPROM.

Table H-4. Interrupt Vectors for Z180 Internal Devices

Address	Name	Description
0x00	INT1_VEC	Expansion bus attention INT1 vector.
0x02	INT2_VEC	INT2 vector.
0x04	PRT0_VEC	PRT Channel 0
0x06	PRT1_VEC	PRT Channel 1
0x08	DMA0_VEC	DMA Channel 0
0x0A	DMA1_VEC	DMA Channel 1
0x0C	CSI/O_VEC	Clocked Serial I/O
0x0E	SER0_VEC	Asynchronous Serial Port Channel 0
0x10	SER1_VEC	Asynchronous Serial Port Channel 1

A directive such as the following is used to “vector” an interrupt to a user function in Dynamic C.

```
#INT_VEC 0x10 myfunction
```

This statement causes the interrupt at offset 0x10 (Serial Port 1 of the Z180) to invoke the function `myfunction()`. The function must be declared with the `interrupt` keyword as follows.

```
interrupt myfunction() {  
    ...  
}
```

Interrupt Priorities

Table H-5 lists the interrupt priorities from highest to lowest.

Table H-5. Interrupt Priorities

Interrupt Priorities	
(Highest Priority)	Trap (Illegal Instruction)
	NMI (Nonmaskable Interrupt)
	INT 0 (Maskable interrupts, Level 0, 3 modes, PIO interrupts)
	INT 1 (Maskable interrupts, Level 1. PLCBus attention line interrupt)
	INT 2 (Maskable interrupts, Level 2)
	PRT Channel 0
	PRT Channel 1
	DMA Channel 0
	DMA Channel 1
	Clocked Serial I/O
	Asynchronous Serial Port 0
(Lowest Priority)	Asynchronous Serial Port 1

Blank



*APPENDIX I: **BATTERY***

Appendix I provides information about the onboard lithium battery.

Storage Conditions and Shelf Life

The lithium battery will provide approximately 9,000 hours of backup time for the onboard real-time clock and static RAM. However, backup time longevity is affected by many factors including the amount of time the PK2500 is unpowered. Most systems are operated on a continuous basis, with the battery supplying power to the real-time clock and the SRAM during power outages and/or during routine maintenance. The time estimate reflects the shelf life of a lithium ion battery with occasional use rather than the ability of the battery to power the circuitry full time.

The battery has a capacity of 165 mA·h. At 25°C, the real-time clock draws 3 μ A when idle, and the 128K SRAM draws 4 μ A. If the PK2500 were unpowered 100 percent of the time, the battery would last 23, 570 hours (2.7 years).

To maximize the battery life, the PK2500 should be stored at room temperature in the factory packaging until field installation. Take care that the PK2500 is not exposed to extreme temperature, humidity, and/or contaminants such as dust and chemicals. Replacement batteries should be kept sealed in the factory packaging at room temperature until installation. Protection against environmental extremes will help maximize battery life.

Instructions for Replacing the Lithium Battery

Use the following steps to replace the battery.

1. Locate the three pins on the bottom side of the printed circuit board that secure it to the board.
2. Carefully de-solder the pins and remove the battery. Use a solder sucker to clean up the holes.
3. Install the new battery and solder it to the board. Use only a BR2325-1HG or equivalent.

Battery Cautions

- **Caution (English)**

There is a danger of explosion if battery is incorrectly replaced. Replace only with the same or equivalent type recommended by the manufacturer. Dispose of used batteries according to the manufacturer's instructions.

- **Warnung (German)**

Explosionsgefahr durch falsches Einsetzen oder Behandlein der Batterie. Nur durch gleichen Typ oder vom Hersteller empfohlenen Ersatztyp ersetzen. Entsorgung der gebrauchten Batterien gemäß den Anweisungen des Herstellers.

- **Attention (French)**

Il y a danger d'explosion si la remplacement de la batterie est incorrect. Remplacez uniquement avec une batterie du même type ou d'un type équivalent recommandé par le fabricant. Mettez au rebut les batteries usagées conformément aux instructions du fabricant.

- **Cuidado (Spanish)**

Peligro de explosión si la pila es instalada incorrectamente. Reemplace solamente con una similar o de tipo equivalente a la que el fabricante recomienda. Deshagase de las pilas usadas de acuerdo con las instrucciones del fabricante.

- **Waarschuwing (Dutch)**

Explosiegevaar indien de batterij niet goed wordt vervagen. Vervanging alleen door een zelfde of equivalent type als aanbevolen door de fabrikant. Gebruikte batterijen afvoeren als door de fabrikant wordt aangegeven.

- **Varning (Swedish)**

Explosionsfära vid felaktigt batteribyte. Använd samma batterityp eller en likvärdigt typ som rekommenderas av fabrikanten. Kassera använt batteri enligt fabrikantens instruktion.

Blank

Symbols

#INT_VEC	118
<Ctrl Y>	21, 22
<Ctrl Z>	22
=(assignment)	
use	84

A

A/D converter	
addressing	73
inputs	13, 24, 29, 46
A/D voltage reference	29
advanced I/O programming	70
analog inputs	
configuration	46, 47
frequency response	54
gain calculation	48
gain resistors	47, 48
input impedance	54
input ranges	47
offset resistors	47
offset voltage	52
reading	55
scaling input range	50
scaling inputs	48
signal conditioning	46

B

battery	
cautions	123
replacing	122
baud rate	
reset	22
brownout	62, 97
bufLength256	78
bufPtr	79
bufSize256	79

C

CE compliance	15
changing duty cycles	77
chanNum	35, 64
definition	64
ckalrate	79
clock	
setup	76
common problems	
programming errors	84
communication	
ports	69
Compile	34
icon	22
program	22
configuring	
inputs and outputs	28
serial communications	27
connect PK2500 to PC	18
customization	14

D

default jumper positions	88
digital I/O	64
using	35
digital inputs	13
addressing	71
digital outputs	13
addressing	72
states	72
DMA counter	77
dmapwmBufBeg	79
dmapwmInit	79
dmapwmSetBuf	76, 78
dmapwmSwBuf	79
duty cycle	59
calculating	68
changing	77
Dynamic C	20
serial options	22

E

EEPROM	
simulated	112
EIO_NODEV	64
eioBrdACalib	55
eioBrdAI	55
eioBrdAO	59, 68
eioBrdAORf	68
eioBrdDI	35, 64
eioBrdDO	37, 65
eioBrdInit	55, 64
eioBrdRelay	44
eioErrorCode	64
eioSetupAO1st	64, 68
electrical and mechanical	86
enclosure	
dimensions	108
mounting	109
EPROM	118
establishing communication	20
external dimensions	
PK2500	86
SIB2	105
EZIOPK25.LIB	64, 68

F

F3	22
F9	22
features	13
flash EPROM	
constants	112
writes	
lifetime	34
flash LEDs	22
FLASHLED.C	22
flexibility	14
float	
use	84
function of K	93

H

H1	29
standard configuration	88
H2	
standard configuration	88
H3	
standard configuration	88
H4	
standard configuration	88
H5	29, 42
standard configuration	88
H6	28
standard configuration	88
H7	100
standard configuration	88
halt program	22
header	
H1	29
H5	29, 42
H6	28
H7	100
J4	23
high-current driver	
outputs	13, 24, 28, 37
specifications	92

I

I/O

addresses external to Z180	117
advanced programming	70
combinations	13
configurations	24
functions	24, 23
header J4	23
memory map	114
inputs	
A/D converter	13, 24, 29, 46
A/D voltage reference	29
demonstration program	36
digital	13
pull-up	36

int	
type specifier, use	84
interrupts	118
interrupt routines	97
interrupt service routines	67
priorities	119
vectors	118
ioAddr	79
J	
J1 pin assignments	26, 40
J4 pin assignments	25
jumpers	
default positions	88
standard configuration	88
K	
K	93
L	
LEDs	13, 60
addressing	74
specifications	60
level-sensitive interrupts	66
literal (C term)	
use	84
lithium backup battery	97, 122
M	
mode	
program	32
run	32
standalone	32
multidrop network	42
N	
network	42
newBuf256	79
NMI	62
no-op	77
nonmaskable interrupt	62
O	
offChar	78
operating modes	32
changing	33
permissible activities	33
outputs	
+5 V regulated switching supply	
.....	62
A/D voltage reference	29
channel assignments	65
demonstration program	38, 60
digital	
pulse width modulation	13
high-current drivers ...	13, 28, 37
relay	
maximum switching voltage	43
relays	13, 43
sinking	13
sourcing	13
overload	97
P	
pBufStart	78
permissible activities	33
phyBuffer256	79
pin layout	24
PK2500	
connect to PC	18
default communication rates ...	22
establish communication	20
external dimensions	86
features	13
I/O configurations	24
I/O map	70
network	42
operating modes	32
overview	12
pin layout	24
power-on reset	61
subsystems	30
ports	
serial	14

power	
fail flag	69
SIB2	104
supervisor	61
power failure	
detection circuitry	96
sequence of events	96
power-on	
reset	61
program	
input demonstration	36
mode	32
output demonstration	38, 60
programmable LEDs	13
programming	14
programming cable	18
protected digital inputs ..	13, 24, 35
specifications	90
pulse width modulation	13
PWM	
addressing	74
advanced programming	78
how to use	57
outputs	57, 68
specifications	57
square waves	59
R	
RBOTTOM	47
real-time clock	61, 69
internal registers	116
refresh DMA counter	77
regulated input voltage	97
relay outputs	13, 43
addressing	72
maximum switching voltage ...	43
snubbers	43
suppressing transients ...	43, 44
reset	
adjust baud rate	22
threshold	61
resSize256	79
return to program mode	34
R_g	47
RJ-12 connector	
PK2500	13, 19
SIB2	20, 104
RS-232 communication	40
configuration	27
connector pinouts	40
RS-485 communication	41
configuration	27
maximum number of nodes	40
network	40
termination resistor	42
port	41
termination resistor	43
RS-485 driver	
addressing	74
RTC. <i>See</i> real-time clock	
run	
icon	22
mode	32
program	22
program standalone	34
sample program	22
run/program jumper	19, 33
S	
sample program	
AI.C	56
AO1.C	60, 69
DI.C	36
DO.C	38
PK25FLSH.C	22
PWM output	60
PWM ramp	68
REL.C	45
selecting	
IN-08 and IN-09	29
IN-10 to IN-15	28
OUT-03 to OUT-05	28
OUT-09 to OUT-11	28
RS-485	29
serial communication	24, 40
configuring	27
ports	27, 40

Serial Interface Board 2	20, 104	threshold	
baud rate	104	reset	61
development	20	troubleshooting	
dimensions	105	baud rate	83
power	104	cables	82
serial ports	14	COM port	82, 83
shutdown	97	communication mode	83
SIB2. <i>See</i> Serial Interface Board 2		expansion boards	82
simulated EEPROM	69	grounds	82
sinking drivers	99	operating mode	83
low-side drive	101	power supply	82
software		repeated resets	83
drivers	64	U	
feature reference	69	unregulated input voltage	97
output channel assignments ...	65	user-programmable LEDs	60
source (C term)		V	
use	84	void eioBrdInit	64
sourcing drivers	99	voltage divider	97
high-side drive	102	voltage regulator	
specifications		switching	13
electrical and mechanical	86	W	
high-current driver	92	watchdog	69
LEDs	60	waveform setup	75
protected digital inputs	90	Z	
PWM	57	Z180	
standalone		Serial Port 1	118
mode	32		
run program	34		
state	59		
step	78		
subsystems	30		
T			
termination resistor			
RS-485 communication	43		

Blank



Z-World, Inc.

2900 Spafford Street
Davis, California 95616-6800 USA

Telephone: (530) 757-3737
Facsimile: (530) 753-5141
Web Site: <http://www.zworld.com>
E-Mail: zworld@zworld.com

Part No. 019-0063
Revision D