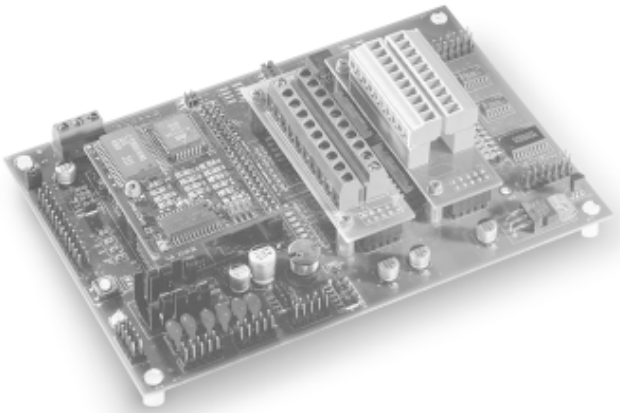


BL1700

C-Programmable Controller

User's Manual

019–0048 • 020415–G

BL1700 User's Manual

Part Number 019-0048 • 020415-G • Printed in U.S.A.

Notice to Users

Z-WORLD PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE-SUPPORT DEVICES OR SYSTEMS UNLESS A SPECIFIC WRITTEN AGREEMENT REGARDING SUCH INTENDED USE IS ENTERED INTO BETWEEN THE CUSTOMER AND Z-WORLD PRIOR TO USE. Life-support devices or systems are devices or systems intended for surgical implantation into the body or to sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling and user's manual, can be reasonably expected to result in significant injury.

No complex software or hardware system is perfect. Bugs are always present in a system of any size. In order to prevent danger to life or property, it is the responsibility of the system designer to incorporate redundant protective mechanisms appropriate to the risk involved.

Trademarks

- Dynamic C® is a registered trademark of Z-World
- Windows® is a registered trademark of Microsoft Corporation
- PLCBus™ is a trademark of Z-World
- Hayes Smart Modem® is a registered trademark of Hayes Microcomputer Products, Inc.

Z-World, Inc.
2900 Spafford Street
Davis, California 95616-6800
USA

Telephone: (530) 757-3737
Facsimile: (530) 757-3792
Web Site: http://www.zworld.com
E-Mail: zworld@zworld.com

TABLE OF CONTENTS

ABOUT THIS MANUAL

This manual provides instructions for installing, testing, configuring, and interconnecting the Z-World BL1700 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 BL1700 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
  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	Electronically 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

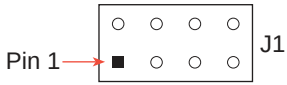
Table 3 lists and defines the typographic conventions that may be used in this manual.

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.
Italics	Indicates that something should be typed instead of the italicized words (e.g., in place of *filename*, 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.

CHAPTER 1: OVERVIEW

Chapter 1 provides an overview and a brief description of the BL1700 features.

Overview

The BL1700 is a feature-rich controller with modular digital and analog I/O that allows easy custom modification. The BL1700 is programmed using Dynamic C, Z-World's version of the C programming language designed for embedded control.

Figure 1-1 illustrates the BL1700 board layout.

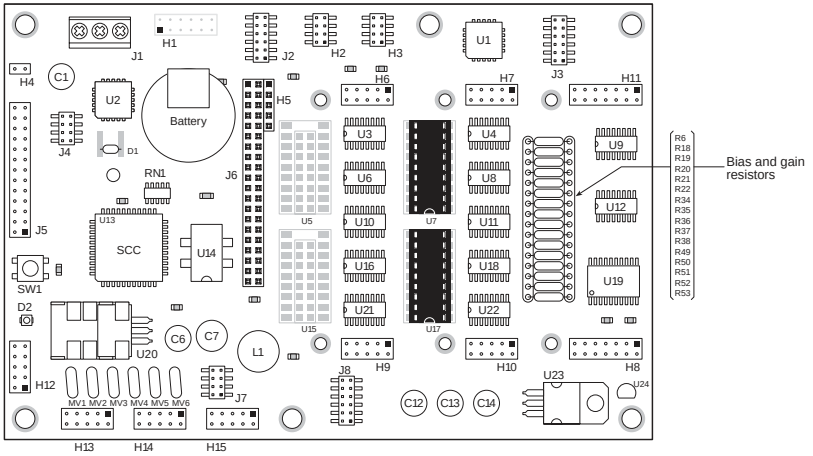


Figure 1-1. BL1700 Board Layout

Features

The BL1700 includes the following features.

• *Core Module*

The BL1700 uses a core module (Z-World part number 129-0099) designed for easy, in-system programming. The core module includes the CPU, RAM, flash EPROM, real-time clock, and microprocessor watchdog circuitry.

• *I/O*

Serial channels—Four full-duplex serial channels interface directly with serial I/O devices. RS-232 and RS-485 signal levels are supported.

Digital inputs—Up to 32 protected digital inputs capable of detecting logic level or high-voltage signals.

Digital outputs—Up to 32 high-voltage, high-current outputs capable of driving resistive and inductive loads.

Pulse-width modulated outputs—Up to 7 digital outputs can provide pulse-width modulation.

Analog inputs—Eight conditioned analog inputs, each with user-configurable bias and gain, interface directly with many sensors. Two unconditioned analog inputs which allow for custom signal conditioning circuitry or direct interfacing.

Expansion bus—I/O expansion via built-in PLCBus. The PLCBus uses inexpensive off-the-shelf Z-World expansion boards.

• *Additional Features*

Field Wiring Terminals—Removable field wiring terminals in several configurations are available for the digital and analog I/O ports.

Compact form factor—Compatible with standard 100 mm wide DIN mounting products.

LED—A general-purpose, user-programmable LED is included.

DIN Rails—The Bl1700 may be mounted in 110 mm DIN rail trays.

Appendix B provides detailed specifications for the BL1700.

See Appendix C, “Field Wiring Terminals (FWT) and DIN Rails,” for more information on FWTs and DIN rail mounting.

Flexibility and Customization

The BL1700 was designed with customization in mind. The design was optimized for cost effective, quick-turn, custom manufacturing. Surface mount technology was used extensively in order to reduce both size and cost while providing the flexibility to meet individual design needs. For quantity orders, the BL1700 can be customized to better meet the needs of your application.

Standard Models

The BL1700 Series of controllers currently has four versions. Table 1-1 lists the standard features for these versions.

Table 1-1. BL1700 Series Features

Model	Features
BL1700	18.432 MHz clock, 16 protected digital inputs, 16 high-voltage sinking outputs, 4 full-duplex serial channels, 10 A/D channels, PLCBus expansion port.
BL1710	BL1700 without A/D channels.
BL1720	BL1700 with two serial channels instead of four.
BL1730	BL1700 with two serial channels instead of four and 9.216 MHz clock.

Customization Options

The BL1700 can be customized for individual applications. The options include the following configurations.

- Core module configuration—CM7100 and CM7200 core modules can be used on the BL1700. Customization options include RAM size, flash EPROM size, EPROM size, clock speed, and real-time clock option.

CM7100 and CM7200 core modules must have a 5-pin header installed at H1, and the BIOS must be customized for these core modules to be used on the BL1700.

- Digital I/O configuration—optional TTL level I/O.
- Analog input configuration—gain and offset configuration.
- Serial channel configuration—two or four serial ports.

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

Development and Evaluation Tools

The BL1700 is supported by a Development Kit that includes everything you need to start development with the BLl700.

Development Kit

The Development Kit includes these items.

- Manual with schematics.
- Programming cables and adapter.
- 24 V DC wall-mount power supply.
- Field wiring terminals.
- Sourcing high-voltage driver ICs.

An optional Serial Interface Board (SIB) allows full access to all serial ports during development.

Software

The BL1700 is programmed using Z-World's Dynamic C, an integrated development environment that includes an editor, a C compiler, and a debugger. Library functions provide an easy and robust interface to the BL1700.

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

CE Compliance

The BL1700 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 BL1700 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.
- Fast transients/burst tests were not performed on the BL1700. Signal and process lines that are longer than 3 m should be routed in a separate shielded conduit.
- The BL1700 has been tested to Light Industrial Immunity standards. Additional shielding or filtering may be required for an industrial environment.
- The BL1700 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.

CHAPTER 2: GETTING STARTED

Chapter 2 provides instructions for connecting the BL1700 to a host PC and running a sample program. The following sections are included.

- Development Kit Packing List
- Connecting the BL1700 to Your PC
- Establishing Communication with the BL1700
- Running a Sample Program

Development Kit Packing List

The BL1700 Development Kit includes the following items.

- Two serial cables with DB-9 and 10-pin header connectors.
- DB-25 to DB-9 serial adapter.
- 24 V DC wall-mount power transformer.
- Two FWT-50 field wiring terminals.
- One FWT-A/D field wiring terminal.
- Two 2985 high-voltage driver ICs.
- BL1700 User's Manual (this document).

Connecting the BL1700 to a Host PC

The BL1700 can be programmed using a PC through an RS-232 port with the programming cable provided in the Developer's Kit. You can also use Z-World's SIB2 to program the BL1700. Using the SIB2 frees all of the serial channels for the application during development. The SIB2 is not part of the standard Developer's Kit, and must be purchased separately. Both programming methods are described below.

For ordering information, call your Z-World Sales Representative at (530) 757-3737.

Connecting the BL1700 to a PC using the serial port.

1. Make sure that Dynamic C is installed on your system as described in the Dynamic C ***Technical Reference*** manual.
2. Connect the 10-pin programming cable from H12 on the BL1700 to the appropriate COM port of your computer as shown in Figure 2-1. Make sure that pin 1 on the ribbon cable connector (indicated by a small triangle on the connector) matches up with pin 1 on H12 (indicated by a small white circle near the corner of the connector).

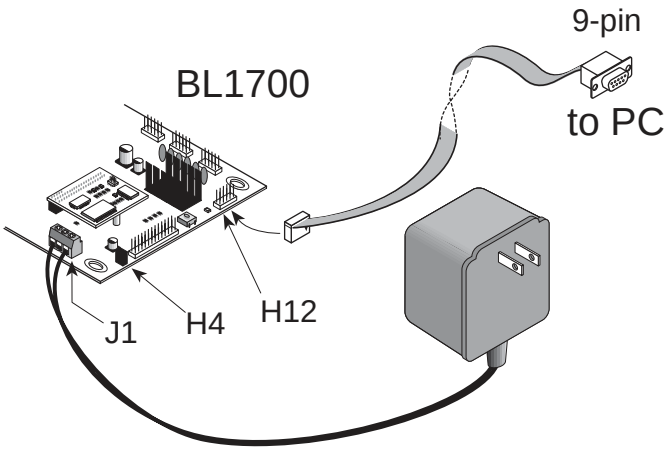


Figure 2-1. BL1700 Programming Connections

Use only the transformer and programming cable supplied by Z-World.

3. Make sure that the Run/Program jumper on header H4 is installed.
4. Connect the 24 V DC transformer as follows.
    - Connect the lead with the red sleeve to the screw terminal (J1) labeled DCIN on the BL1700.
    - Connect the other lead to the screw terminal (J1) labeled GND.
5. Plug the transformer into a wall socket.

Connecting the BL1700 to your PC using the SIB2.

1. Make sure that Dynamic C is installed on your system as described in the ***Dynamic C Technical Reference*** manual.
2. Disconnect power from the BL1700. Connect an RJ-12 cable between the RJ-12/DB-9 adapter attached to the PC and the SIB2.
3. Plug the SIB2's 8-pin connector onto header JP1 located on the CM7200 core module (mounted on the BL1700), as shown in Figure 2-2. Make sure that pin 1 on the ribbon cable connector (on the striped side) matches up with pin 1 on JP1 (indicated by a small white circle next to the header).

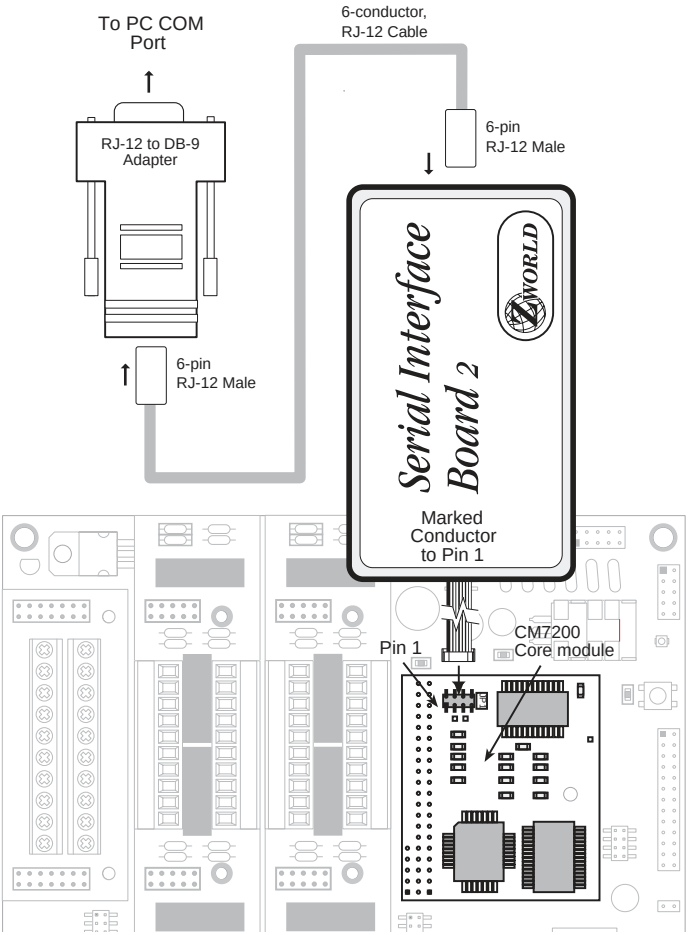


Figure 2-2. SIB2 Connection (BL1700 Top View)

Use only the transformer and programming cable supplied by Z-World.

Observe the polarity of the cable and the 8-pin connector. Attach the connector to JP1 exactly as shown in Figure 2-2.

4. Make sure that the Run/Program jumper on header H4 is installed.
5. Connect the 24 V DC transformer as follows.
    - Connect the lead with the red sleeve to the screw terminal (J1) labeled DCIN on the BL1700.
    - Connect the other lead to the screw terminal (J1) labeled GND.
6. Plug the power supply into a wall socket.

Establishing Communication with the BL1700

1. Double-click the Dynamic C icon to start the software. Note that communication with the BL1700 is attempted each time you start Dynamic C.
2. If the communication attempt is successful, no error messages are displayed.

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 BL1700, use the Dynamic C shortcut **<Ctrl Y>** to reset the controller and initiate communication.

Running a Sample Program

1. Open the sample program **BL17FLSH.C** located in the Dynamic C **SAMPLES\BL17XX** directory. This program flashes the onboard LED.
2. Compile the program by pressing **F3** or by choosing **Compile** from the **Compile** menu. Dynamic C compiles and downloads the program into the BL1700's flash memory.

   During compilation, Dynamic C rapidly displays several messages in the compiling window. This condition is normal.

   See Appendix A, "Troubleshooting," if an error message such as **Target Not Responding** or **Communication Error** appears.
3. Run the program by pressing **F9** or by choosing **Run** from the **Run** Menu.
4. To halt the program, press **<Ctrl Z>**. This action halts program execution.
5. To restart program execution, when required, press **F9**.

CHAPTER 3: BL1700 HARDWARE

Chapter 3 describes the BL1700 hardware subsystems. The following sections are included.

- Operating Modes
- BL1700 Subsystems Overview
- Microprocessor Core Module
- Serial Communications Channels
- High-Voltage Digital Outputs
- Protected Digital Inputs
- Analog Inputs
- PLCBus Expansion Port

Operating Modes

The BL1700 has two mutually exclusive operating modes, run mode and program mode. Each mode is explained in detail below.

- **Program Mode**

  In program mode, the BL1700 controller runs under the control of your PC that is running Dynamic C. The BL1700 must be in this mode to compile a program to the BL1700 or debug a program.

  - In program mode, the BL1700 matches the baud rate of the PC COM port up to 57,600 bps.
  - USER LED is "ON" in program mode.

- **Run Mode**

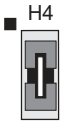
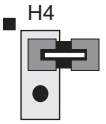
  In run mode, the BL1700 controller runs standalone. At power-up, the BL1700 checks to see if its onboard memory contains a program. If a program exists, the BL1700 controller executes the program immediately after power-up.

  - In run mode, the BL1700 does not respond to Dynamic C running on the PC. A program cannot be compiled or debugged when the BL1700 is in run mode.
  - USER LED D2 is under the control of the application on the BL1700 when the BL1700 is in run mode.

Table 3-1 shows the jumper settings for the program and run modes.

Table 3-1. BL1700 Jumper Settings for Run/Program Modes

Operating Mode	Header H4	Permissible Activities
Program Mode		• Compile a program. • Run a program under debugger control. • Run a program without "polling." See your Dynamic C manuals for a description of program polling.
Run Mode		Run application.

Changing the Operating Mode

1. Locate the **Run/Program** jumper on header H4. Figure 3-1 shows the location of header H4.

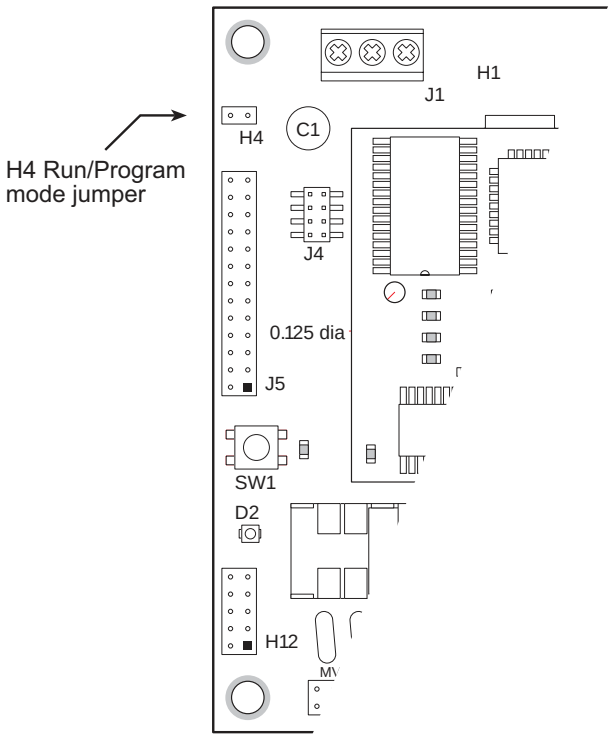


Figure 3-1. H4 Run/Program Jumper Location

2. Select the desired operating mode.
   - Install jumper on header H4 to select program mode.
   - Remove jumper on header H4 to select run mode.
3. Press the reset switch SW1 to switch the BL1700 to the selected mode.

> Be sure careful when installing or removing the H4 jumper if power is connected to the BL1700.

Run Mode

1. Place the BL1700 in program mode (with the H4 jumper installed) 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** on your keyboard.
4. If no errors are detected, Dynamic C compiles the program and automatically downloads it into the BL1700's onboard flash memory.
5. Remove the **Run/Program** jumper.
6. Press the reset switch SW1 on the BL1700. This action resets the BL1700 and places it into run mode. The downloaded program begins to run immediately.

The downloaded program begins to run as soon as the reset switch is pressed or power is applied. Pay close attention to any electronic or mechanical devices connected to the BL1700 that could cause injury.

The program is now loaded in the BL1700's onboard flash EPROM. This program runs automatically every time the BL1700 powers up in run mode until you load another program.

Follow these steps to return to the program mode.

1. Re-install the **Run/Program** jumper on header **H4**. Refer to Figure 3-1 for the jumper location.
2. Press the reset switch on the BL1700.

Refer to the previous section, "Changing the Operating Mode," for more detailed information.

BL1700 Subsystems Overview

The BL1700 is comprised of several subsystems including a microprocessor core module, serial communications channels, digital I/O, analog inputs, and PLCBus expansion port. Figure 3-2 illustrates the BL1700 subsystems.

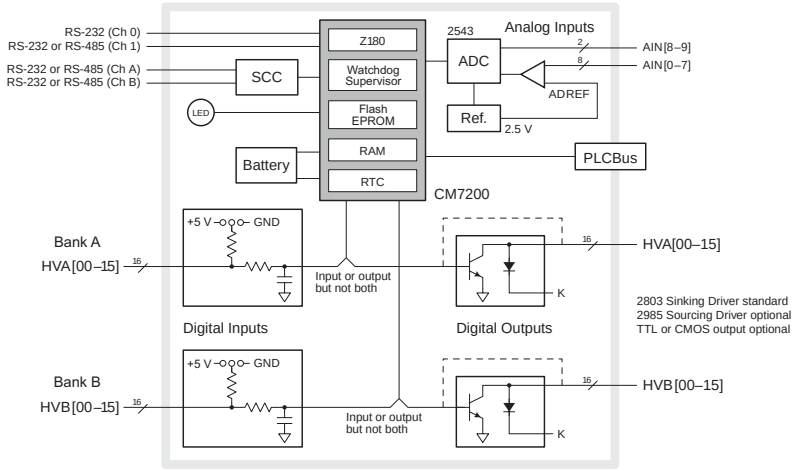


Figure 3-2. BL1700 Block Diagram

Microprocessor Core Module

The BL1700 is built around a Z-World CM7200 Series microprocessor core module. The core module is comprised of a Zilog Z180 microprocessor, 32K of battery-backed static RAM, 128K of flash EPROM, a real-time clock, and a watchdog timer/microprocessor supervisor.

The Z180 CPU runs at 18.432 MHz. Internal to the Z180 are two asynchronous serial ports, two DMA channels, two programmable-reload timers (PRTs), and three interrupt lines.

Six chip-select lines (/CS1–/CS6) enable one of six groups of 64 I/O addresses. These lines are used to access peripherals on the BL1700 board.

The power-supervisor IC performs several functions. It provides a watchdog timer function, performs power-failure detection, RAM protection, and battery backup when the CM7200 is unpowered.

Your program can obtain the time and the date from the real-time clock.

Figure 3-3 shows a block diagram of the CM7200 microprocessor core module.

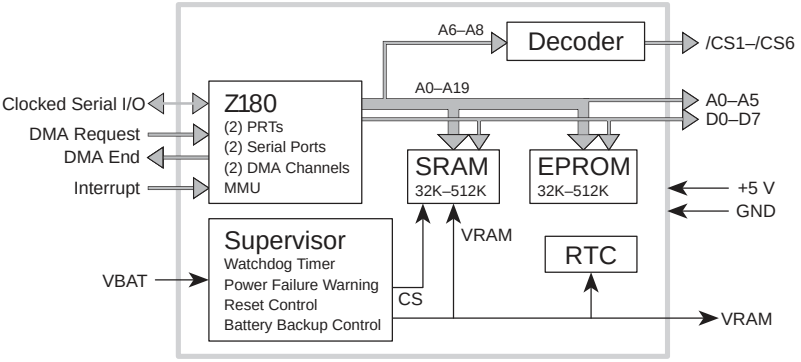


Figure 3-3. CM7200 Block Diagram

Core Module External Connections

The core module also provides connections to the Clock Serial I/O (CSIO) port on the Z180. This port can be used to program the BL1700 using Z-World's Serial Interface Board 2 (SIB2). This allows programming and debugging of the BL1700 while providing access to all the onboard serial channels.

Digital Inputs and Outputs

The digital inputs and outputs are divided into two banks, A and B, as shown in Figure 3-2 and Figure 3-4. The 16 factory-default digital inputs on the BL1700, BL1710, BL1720, and BL1730 occupy Bank A, and 16 digital outputs are located on Bank B. Future and/or custom versions of the BL1700 may have both or no banks configured as digital inputs. In order for a bank to be configured as an input, the appropriate interface ICs must be installed. In order for a bank to be configured as an output, the appropriate high-voltage driver ICs must be installed. These modifications should only be performed at Z-World's manufacturing facility.

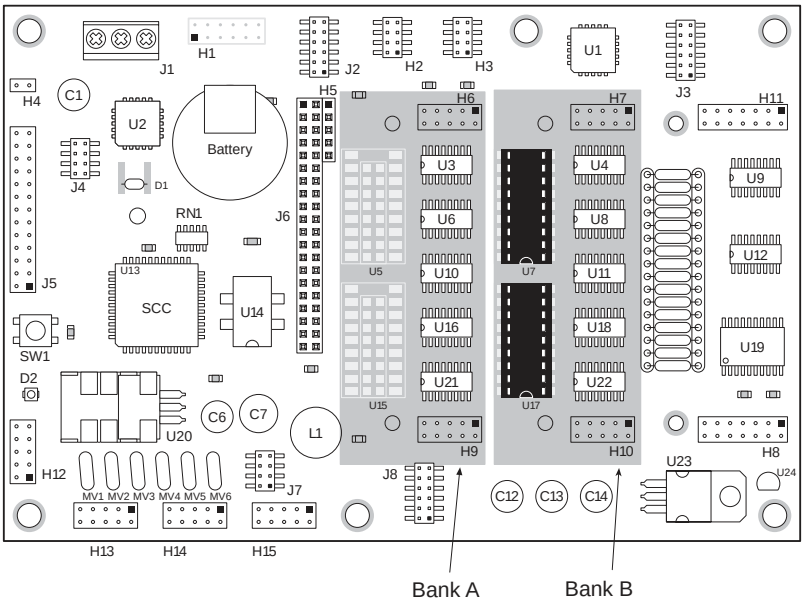


Figure 3-4. BL1700 Banks A and B

External Connections

Connections to Bank A are made on headers H6 and H9. Connections to Bank B are made on headers H7 and H10. The pinouts for headers H6, H7, H9 and H10 are shown in Figure 3-5.

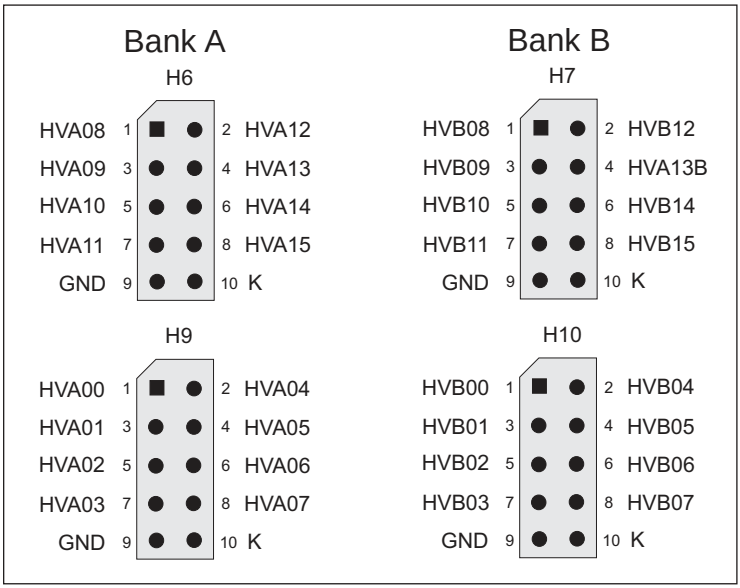


Figure 3-5. Pinouts for BL1700 Digital Input External Connections

Connections to the digital inputs/outputs can be made with a ribbon cable, Z-World's FWT field wiring terminals, or a custom interface board. Z-World offers FWT modules for the digital inputs in three configurations.

- Screw terminals (Z-World part number 101-0184).
- Removable screw terminals (Z-World part number 101-0185)
- Optically isolated removable screw terminals (Z-World part number 101-0186)

Input lines connected to optically isolated devices must be configured as pull-up. Otherwise, damage to the circuit may occur.

Each FWT module mates with one of the BL1700's header pairs (H6–H9 and H7–H10). Different types of field wiring terminals can be mixed on the same BL1700.

See Appendix C, "Specifications," for FWT mechanical dimensions and pinouts.

Digital Inputs

The BL1700 can provide up to 32 protected digital inputs designed as logical data inputs, returning a 1 or 0. Their normal operating range is -20 V DC to +24 V DC, and they are protected from voltages between -48 V DC and +48 V DC. The inputs can detect logic-level signals and have a nominal logic threshold of 2.5 V DC. This means an input returns a 0 if the input voltage is below 2.5 V DC and a 1 if the input voltage is above 2.5 V DC. The inputs can be pulled up to +5 V or down to ground.

A low-pass filter on each input channel has a time constant of

T_{RC} = 220 µs (4.5 kHz).

They may be configured as pull-up or pull-down in groups of fours and eights. The configuration of each input should be determined by normal operating conditions, power-down mode and possible failure modes including open or shorted conditions. These factors will influence your decision about configuring the inputs as pull-up or pull-down.

Operating Modes and Configuration

Inputs may be pulled up to +5 V or pulled down to ground by configuring the jumpers on BL1700 headers J2 and J3.

J2 jumpers select pull-up/pull-down resistors for Bank A. Jumpers on J3 select pull-up/pull-down resistors for inputs for Bank B. To change an input from the factory default of pull-up, simply place a jumper across the appropriate two pins of J2 and/or J3.

Table 3-2 and Table 3-3 illustrate the jumper settings for pull-up and pull-down configurations for the BL1700's Bank A and Bank B inputs.

The factory default is for the digital inputs to be pulled up to +5 V.

Table 3-2. BL1700 Bank A Digital Input Jumper Configurations

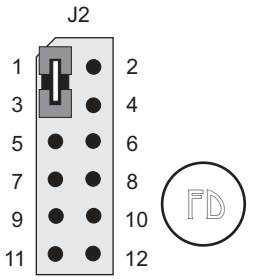
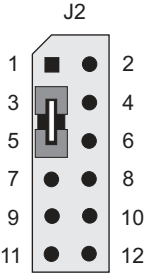
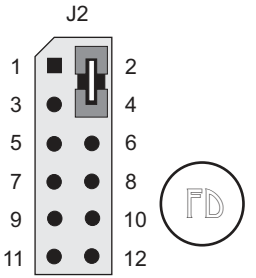
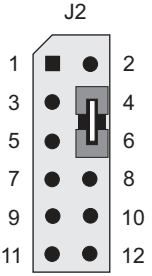
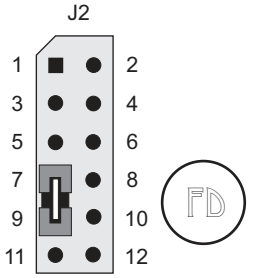
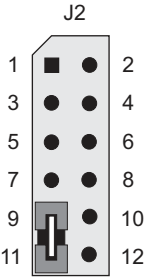
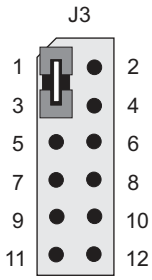
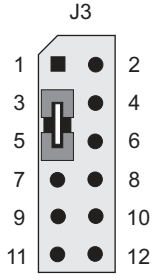
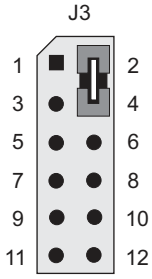
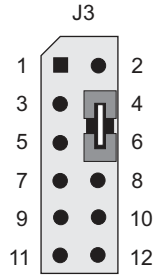
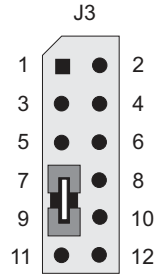
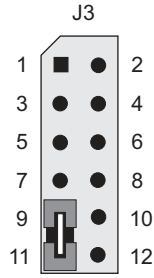
Channel	Jumper Settings	
	Inputs Pulled Up	**Inputs Pulled Down**
HVA 8–11 Bank A Channels 8–11 (Physical Channels 24–27)		
HVA 12–15 Bank A Channels 12–15 (Physical Channels 28–31)		
HVA 0–7 Bank A Channels 0–7 (Physical Channels 16–23)		

Table 3-3. BL1700 Bank B Digital Input Jumper Configurations

Channel	Jumper Settings	
	Inputs Pulled Up	**Inputs Pulled Down**
HVB 0–3 Bank B Channels 0–3 (Physical Channels 0–3)		
HVB 4–7 Bank B Channels 4–7 (Physical Channels 4–7)		
HVB 8–15 Bank B Channels 8–15 (Physical Channels 8–15)		

The high-voltage driver chips must be removed from Bank B and interface chips must be installed before the Bank B inputs can be used as digital inputs.

Digital Outputs

Up to 32 high-voltage, high-current digital outputs are possible on the BL1700. The digital outputs can be configured in groups of eight for either sinking or sourcing operation by setting jumpers and installing the appropriate driver ICs. Sinking drivers can sink up to 500 mA at voltages up to 48 V DC. Sourcing drivers can source up to 250 mA at voltages up to 30 V DC. All outputs are diode protected against inductive spikes.

TTL/CMOS level outputs are also possible by bypassing the driver ICs. This option is for quantity orders only, and should be performed at Z-World's manufacturing facility.

High-voltage outputs are diode protected against inductive spikes. All outputs are individually addressable.

Operating Modes and Configuration

The digital inputs and outputs are divided into two banks, Bank A and Bank B. In the factory default, digital outputs occupy Bank B and digital inputs are located on Bank A. In order for a bank to be configured as an output, the appropriate interface ICs must be installed. Z-World recommends that this be done only at Z-World's manufacturing facility.

High-Voltage Drivers

Outputs may be configured for either sinking or sourcing current. The configuration is determined by the type of driver ICs installed and the jumper settings.

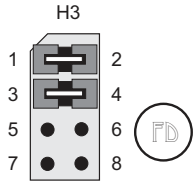
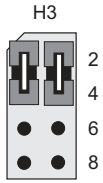
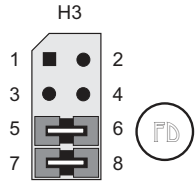
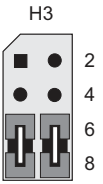
For Bank A, U5 drives outputs 8-15 and U15 drives outputs 0-7. For Bank B, U7 drives outputs 8-15 and U17 drives outputs 0-7. The jumpers placed on H3 configure sourcing/sinking modes for the outputs on Bank B. Jumpers on H2 configure sourcing/sinking modes for the outputs on Bank A (if it is configured for output). Table 3-4 and Table 3-5 show the jumper settings for sinking and sourcing configurations.

The sinking driver chips used on the BL1700 are ULN2803 or equivalent. The sourcing driver chips are UDN2985 or equivalent.

To configure drivers for sinking outputs (default for Bank B), install the ULN2803 driver chips in the appropriate socket locations. For sourcing outputs, install UDN2985 driver chips.

When installing high-voltage driver chips, make sure that pin 1 on the IC matches up with pin 1 on the socket. The chip has a small semicircular notch on one end that matches up with a similar notch on the IC socket. The chips can be removed by gently prying them out with a small screwdriver or IC extractor.

Table 3-4. BL1700 Bank B Digital Output Jumper Configurations

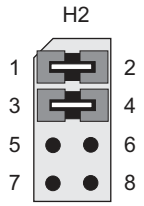
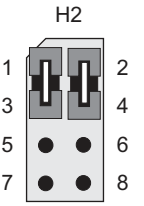
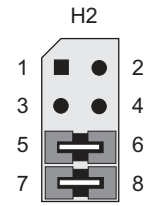
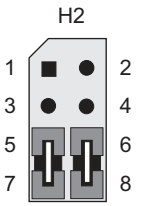
Bank B	Jumper Settings	
	Sinking Outputs	Sourcing Outputs
HVB 0–7 Channels 0–7	 U17 = ULN2803	 U17 = UDN2985
HVB 8–15 Channels 8–15	 U7 = ULN2803	 U7 = UDN2985

Also make sure that the jumpers on H3 and/or H2 are properly configured. If the jumpers are not properly set for the drivers installed, damage to both the drivers and the circuit board is possible.

Connections to Bank A are made on headers H6 and H9. Connections to Bank B are made on headers H7 and H10. The pinouts for headers H6, H7, H9 and H10 are shown in Figure 3-5 on page 30.

See Appendix B, “Specifications,” for detailed specifications on the high-voltage drivers.

Table 3-5. BL1700 Bank A Digital Output Jumper Configurations

Bank A	Jumper Settings	
	Sinking Outputs	**Sourcing Outputs**
HVA 8–15 Channels 8–15	 U5 = ULN2803	 U5 = UDN2985
HVA 0–7 Channels 0–7	 U15 = ULN2803	 U15 = UDN2985

The digital interface chips must be removed from Bank A and high-voltage driver chips must be installed before the Bank A inputs can be used as outputs.

Pulse-Width Modulation (PWM) Configuration

In order to use the PWM feature of the digital outputs, J8 must be jumpered from pin 4 to pin 6. See Figure 3-6.

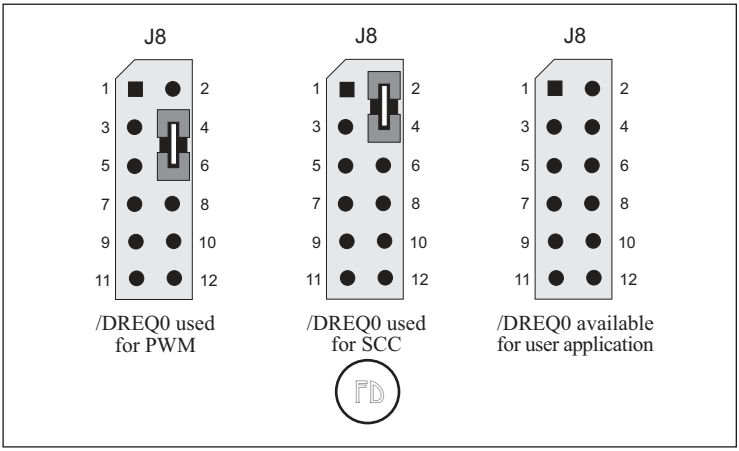


Figure 3-6. /DREQ0 Jumper Settings

Analog Inputs

The BL1700, BL1720, and BL1730 provide 10 single-ended analog-to-digital conversion channels with 12-bit resolution. Eight channels are conditioned and two are unconditioned. The eight conditioned inputs can measure bipolar or unipolar signals. User-installable resistors determine the signal conditioning for your application. Two inputs are connected directly to the A/D converter.

The BL1710 does not have analog inputs.

Operating Modes and Configuration

User-selected gain and bias resistors determine voltage ranges for the conditioned input signals.

Standard BL1700, BL1720, and BL1730 controllers come with 2370 Ω gain resistors and 39.2 kΩ bias resistors. These resistors provide a gain of 0.25 for a unipolar input signal range of 0 V to 10 V.

The BL1700 comes with gain and bias resistors installed for an input range of 0 V to 10 V. Table 3-6 lists the gain and bias resistors for other selected input-voltage ranges. A step-by-step procedure follows to explain how to calculate the values for the gain and bias resistors for a particular input-voltage range.

Table 3-6. Representative Analog Input Setups

Input Voltage Range (V)	Gain	R_{gain} (Ω)	R_{bias} (Ω)
-10.0 to +10.0	0.125	1180	8060
-5.0 to +5.0	0.250	2370	6650
-2.5 to +2.5	0.500	4750	4990
-2.0 to +2.0	0.625	5900	4530
-1.0 to +1.0	1.250	11,800	2870
-0.5 to +0.5	2.500	23,700	1690
-0.25 to +0.25	5.000	47,500	931
-0.10 to +0.10	12.500	118,000	392
0 to + 10.0	0.250	2370	39,200
0 to +5.0	0.500	4750	20,000
0 to +2.5	1.000	9530	10,000
0 to +1.0	2.500	23,200	4020

1. Set up the analog inputs.

The first eight analog input signals are routed to the inverting input of one of the eight op-amps in U9 and U12. The op-amps in U9 and U12 operate in an inverting configuration. User-selectable resistors set the gain and bias voltages of the amplifiers. The 10 kΩ input resistors are fixed. Feedback capacitors roll off the high-frequency response of the amplifiers to attenuate noise. Figure 3-7 shows a schematic diagram of the conditioned input amplifier circuit.

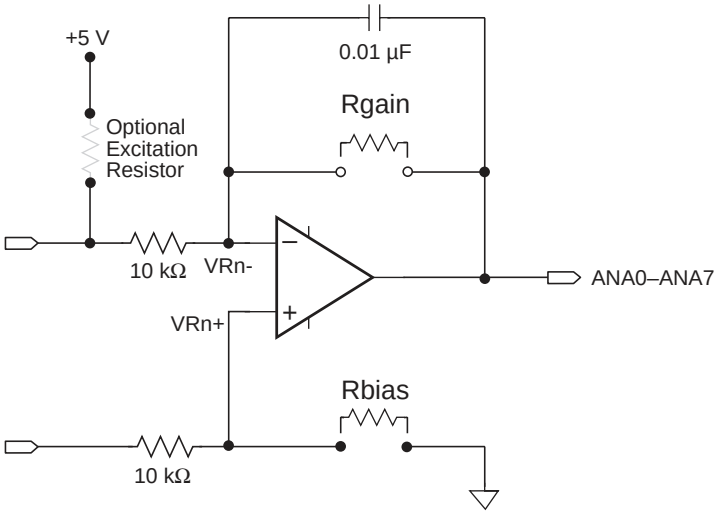


Figure 3-7. Analog Conditioning Circuit

Table 3-7 lists the gain and bias resistors for each of the eight conditioned analog input channels.

Table 3-7. Gain and Bias Resistors

Channel	R_{bias}	R_{gain}
ANA0	R20	R21
ANA1	R19	R34
ANA2	R6	R22
ANA3	R18	R35
ANA4	R51	R36
ANA5	R52	R49
ANA6	R53	R37
ANA7	R50	R38

Strip sockets spaced 0.400 inches (10.2 mm) apart accommodate the gain and bias resistors.

Z-World can install surface-mounted excitation, gain and bias resistors for your exact configuration in production quantities. For more information, call your Z-World Sales Representative at (530) 757-3737.

2. Select gain resistor.

The gain and bias resistors determine the input signal's voltage relative to ground as well as its range. For example, assume your circuit must handle an input signal voltage range of 10 V spanning -5 V to +5 V. You should first select the gain (feedback) resistor to suit an input signal voltage range of 10 V.

The gain of the amplifier is the ratio of its maximum output-voltage swing to your application's maximum input-voltage swing. The 2.5 V input-voltage range of the A/D chip limits the op-amp's output swing to 2.5 V. Therefore, Equation (3-1) expresses an amplifier's gain in terms of its input-voltage range.

$$g = \frac{2.5\ V}{V_{IN_{max}} - V_{IN_{min}}} \qquad (3\text{-}1)$$

where g is the gain, $V_{IN_{max}}$ is the maximum input voltage and $V_{IN_{min}}$ is the minimum input voltage.

The ratio of the user-specified gain resistor R_{gain} to its associated fixed input resistor determines an amplifier's gain. For the amplifier in Figure 3-7 with its input resistor fixed at 10 kΩ, the gain is

$$g = \frac{R_{gain}}{10{,}000\ \Omega} \quad . \qquad (3\text{-}2)$$

Given an input voltage range of 10 V, this gain equation fixes the amplifier's gain at 0.25. This gain scales the input signal's range properly down to the op-amp's 2.5 V maximum output range. R_{gain} must therefore be 2500 Ω.

3. Determine bias resistor.

If the op-amp is to servo its output properly around the desired center voltage, you must establish the appropriate bias voltage at the op-amp's noninverting input. You must select the bias, or offset, resistor, R_{bias}, to position the input-voltage range correctly with respect to ground. For this example, let us use -5 V to +5 V.

Because the value for R_{gain} has already been selected, the maximum input voltage, V_{INmax}, determines the maximum voltage seen at the amplifier's summing junction (inverting input)—circuit nodes VR0– through VR7–. Compute VR0– through VR7– using Equation (3-3).

$$VR0 = V_{IN_{max}} \times \left(\frac{g}{1+g} \right) \tag{3-3}$$

For each op-amp, the bias voltage, V_{bias}, must equal its corresponding VRn–. A voltage divider, comprising a bias resistor and a fixed 10 kΩ resistor, derive the bias voltage from VREF+. Note that VREF+ is not necessarily the same as REF+. REF+ is the positive reference voltage the A/D chip uses.

VREF+ is 2.5 V and R_{bias} is

$$R_{bias} = \frac{V_{bias} \times 10{,}000\ \Omega}{2.5\ V - V_{bias}} \quad . \tag{3-4}$$

Continuing the example for an input-voltage range that necessitates a gain of 0.25, and for which V_{MAX} is +5 V, V_{bias} is then 1.0 V. Therefore, R_{bias} is 6667 Ω in absolute mode.

Now suppose that the input range is 0 V to +10 V instead of –5 V to +5 V. V_{max} is now +10 V and V_{bias} becomes 2.0 V. R_{bias} is then 40 kΩ.

4. Choose resistor values.

The calculated values, of course, will not always be available as standard resistor values. In these cases, use the nearest standard resistor value. For example, rather than 6667 Ω, use 6650 Ω if you are using 1% resistors, or use 6800 Ω if you are using 5% resistors.

5. Bracket input range.

To be sure of accurately measuring signals at the extremes of an input range, you must be aware of the interaction between the 10 kΩ fixed resistors and the resistors you install. In the ideal case, if you were to measure a signal at the minimum input level, the A/D converter's input would be at the maximum expected value of 2.5 V.

However, in the real world, resistor values vary within their rated tolerance bands. Thus, if the fixed input resistor is lower than its nominal value, and the installed resistor is slightly higher than its nominal value, the actual input to the A/D converter would be greater than 2.5 V. A loss of accuracy then results because the A/D converter input would reach its maximum input value before the true signal input reaches the minimum expected input level, as shown in Figure 3-8.

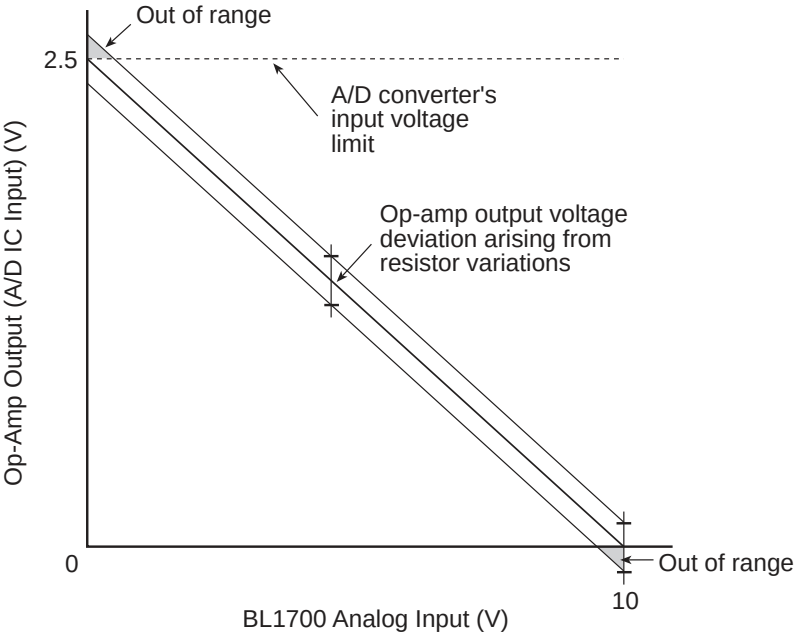


Figure 3-8. Input Out of Range

A deviation from nominal values in the bias network could skew the A/D converter's input voltage away from the theoretically computed value. For example, a small positive or negative deviation of the bias voltage arising from variances in the resistive divider would offset the A/D converter's input voltage. This offset would be positive or negative, tracking the deviation's sign, and would be equal to the bias deviation multiplied by the amplifier's gain plus one. Both of these effects could occur in the same circuit.

6. Pick proper tolerance.

Use care when compensating for any discrepancies discovered. For example, if you use standard 5% resistors, the values are spaced approximately 10% apart. If your gain is too high by just a small amount, then going to the next smallest standard 5% value could result in a drop in gain, and an A/D converter excursion approaching 10%. The same caveat applies to the bias network. Using 1% resistors allows a more precise choice of values.

Figure 3-9 illustrates the result of adjusting the resistor values so that the input signal to the A/D converter stays within its specified 2.5 V range.

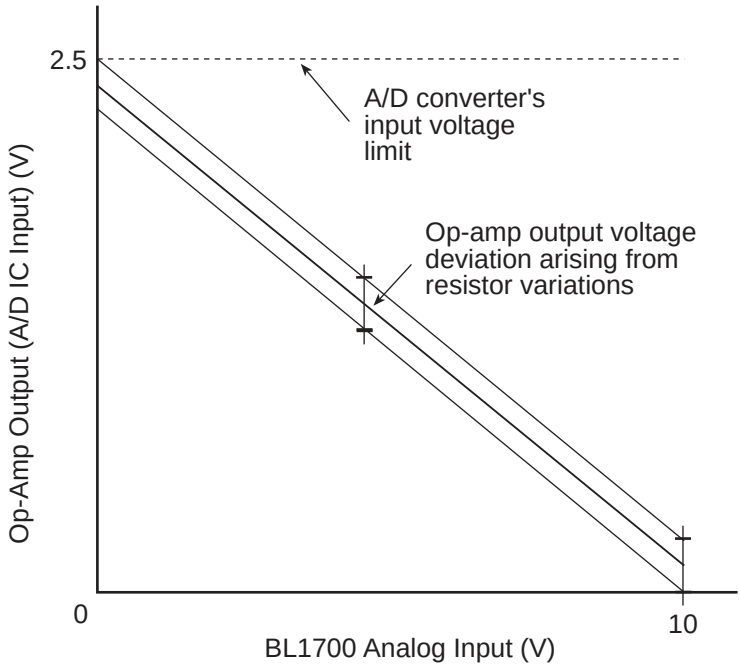


Figure 3-9. Proper Input Range

7. Confirm performance.

If your measurements are critical, check setups after installing resistors by measuring test signals at and near the input-voltage limits. See if the voltages fall within the A/D converter's input range or if accuracy is lost due to over-excursions at the A/D converter's input. Another method is to measure the resistance of the factory-installed fixed resistors before selecting your own resistors.

You can indirectly measure the fixed resistors after installation by measuring the voltages at the amplifiers' inputs and outputs. See Figure 3-10.

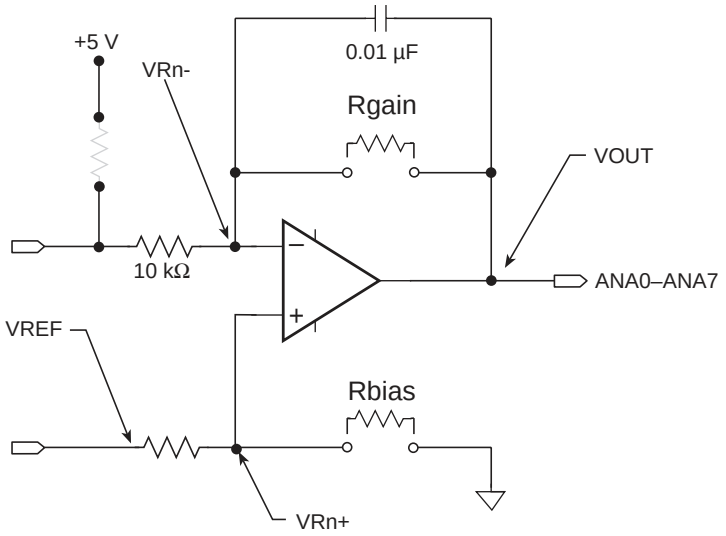


Figure 3-10. Signal Conditioning Test Points

Using Channel 0 as an example, ground the input A0 at pin 1 of H11. Then measure the voltages at VR0- and the amplifier's output. Because the currents through the input resistor and the feedback resistor are essentially identical, the ratio of the voltages across the resistors is equivalent to the ratio of the resistors. Therefore, the gain is

$$\text{gain} = \frac{\text{VOUT} - \text{VR0}-}{\text{VR0}-} \quad . \tag{3-5}$$

Again using Channel 0 as an example, measure the voltage of VREF and the voltage at VR0+. Because the current into the op-amp input is negligible, the resistance ratio of the two resistors in the voltage divider alone determines VR0+. You can then compute the value of the fixed resistor in the divider once you know both the value of the resistor you installed and the value of VR0+.

8. Calibrate the BL1700 A/D converter.

Mathematically derived values provide good baseline gain values. Calibration is necessary because the inherent component-to-component variations of resistors can completely swamp the 0.25% resolution of the A/D converter. To achieve the highest accuracy possible, calibrate the BL1700.

Dynamic C provides a routine to compute calibration coefficients and store the coefficients in nonvolatile memory. The routine uses two reference points to compute the coefficients. Each reference point comprises a pair of values: the actual applied test voltage and raw converted A/D value (a 12-bit integer). The supplied Z-World A/D software will automatically use these calibration coefficients to correct all subsequent A/D readings.

The factory installed fixed resistors have a 1% tolerance.

Calibration constants for the factory installed resistors are stored in simulated EEPROM during testing.

9. Recalibrate the BL1700.

To recalibrate a BL1700, apply two known test voltages to each channel you plan to use. Get the converted reading for each test voltage and pass them, along with the test voltages, to the function **eioBrdACalib** to calculate the conversion coefficients for that channel. **eioBrdACalib** will automatically store the coefficients in the flash EPROM.

Sample program **BL17AIN.C** in the Dynamic C **SAMPLES** directory shows how to calibrate the conditioned analog input channels of a BL1700 manually, assuming test voltages of 1.00 V and 9.00 V.

Drift

The AD680JT voltage reference displays a voltage drift of 10 ppm/°C (typ) to 30 ppm/°C (max). This drift corresponds to 25 mV/°C to 75 mV/°C, or 1.75 mV to 5.25 mV over the temperature range of 0°C to 70°C.

The LMC660C operational amplifier exhibits an offset-voltage drift of 1.3 μV/°C (typ), or 91 mV over the operating temperature range.

Low-Pass Filter

The 0.01 mF feedback capacitors in the amplifier's feedback path transform the amplifiers into low-pass filters. These filters attenuate any high-frequency noise that may be present in your signal. These filters' characteristics depend on the resistors your select.

The 3 dB corner frequency of a filter is

$$f_{3\,db} = \frac{1}{2\pi \times R_g \times 0.01\,\mu F} \quad . \tag{3-6}$$

For the case above with a gain of 0.25 using a 1% feedback resistor of 2490 Ω, the 3 dB corner frequency is 6392 Hz.

Excitation Resistors

Some transducers require an excitation voltage. For example, a thermistor, serving as one leg of a voltage divider (having a fixed resistor in the other leg), measures temperature. The voltage at the divider's junction will vary with temperature. There is provision for excitation resistors to be installed on the inputs of the eight conditioned analog channels. The excitation resistors are tied to the +5 V analog supply.

Using the Unconditioned Converter Channels

The eight conditioned channels use the first eight channels, AIN0–AIN7, of the A/D converter chip. Two additional channels are also available. You can access these channels with software by inserting your desired channel number in the library functions that control the BL1700. These signals are available on headers H8 and H11.

For optimum results, drive these channels with low output impedance voltage sources–less than 50 Ω. Op-amps are ideal for this purpose. High output impedance sources, on the other hand, are susceptible to coupled noise. In addition, only a low-impedance source can quickly charge the sampling capacitors within the A/D converter. When designing the signal sources to drive the extra channels, be sure to consider whether the amplifiers you choose can handle the capacitance of the cable that connects to the analog input connectors.

Internal Test Voltages

In addition to the external input channels of the A/D converter chip, three additional internal channels exist to measure reference points within the A/D converter chip. Unfortunately, the A/D converter compares its internal nodes to REF+ and REF- so the conversions yield either all 1s or all 0s. You may access these channels using ordinary library routines by specifying the appropriate channel address when calling the functions.

Table 3-8. Internal Test Voltages

Channel	Internal Voltage Read
Channel 11	(VREF+ – VREF–) ÷ 2
Channel 12	VREF–
Channel 13	VREF+

Power-Down Mode

If you select Channel 14, the A/D converter chip enters a power-down mode in which all circuitry within the chip goes into a low-current, standby mode. Upon power-up and before the first conversion, the chip also goes into the power-down mode. The chip remains in the power-down mode until you select a channel other than 14. The normal operating current of the A/D converter chip is 1 mA to 2.5 mA. In power-down mode this consumption is reduced to 4 µA to 25 µA.

External Connections

Connections to the analog inputs can be made with a ribbon cable, Z-World's FWT field wiring terminals, or a custom interface board. Z-World offers FWT modules for the digital inputs in three configurations.

- Screw terminals (Z-World part number 101-0184).
- Removable screw terminals (Z-World part number 101-0185)

The FWT module mates with the BL1700's header pairs H8–H11.

Connections to the analog inputs are made on headers H8 and H11. The pinouts for headers H8 and H11 are shown in Figure 3-11.

See Appendix C, "Field Wiring Terminals and DIN Rails," for FWT mechanical dimensions and pinouts.

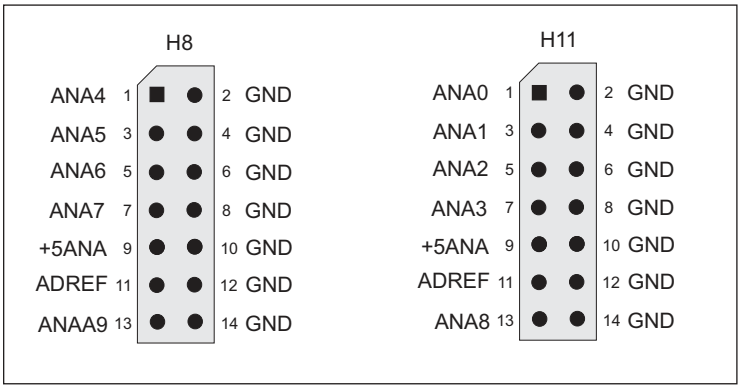


Figure 3-11. Pinouts for BL1700 Analog Input Headers H8 and H11

Serial Channels

Four serial channels are available on the BL1700. One channel, Channel 0, is a dedicated RS-232 communication channel. The other three are available in either RS-232 or RS-485 configurations. Channel 0 and Channel 1 are connected to the Z180's Serial Channel 0 and Serial Channel 1, respectively. Channel A and Channel B are controlled by the Serial Communications Controller (SCC) chip on the BL1700; these two ports also have hardware support for synchronous communication. Serial channel signals are routed to either RS-232 or RS-485 converters via configuration jumpers. Baud rates up to 57,600 bps are supported.

The BL1720 and BL1730 versions have two serial ports. The serial ports on the BL1720–BL1730 versions do not support synchronous communication.

Table 3-9 summarizes the operating modes for the four channels.

Table 3-9. Serial Channel Configuration Options

Channel	Configurations
Channel 0	Three-wire or five-wire RS-232 only
Channel 1	Two-wire RS-485 or three-wire RS-232
Channel A	Two-wire RS-485 or five-wire RS-232, plus DCD and DTR
Channel B	Two-wire RS-485 or five-wire RS-232

Channel 0

Channel 0 is the BL1700's RS-232 programming port and is configured as three-wire or five-wire RS-232. Channel 0 cannot be reconfigured.

Channel 1

Channel 1 is a general-purpose serial channel that can be configured as two-wire RS-485 or three-wire RS-232.

Channel A

Channel A is a general-purpose serial channel controlled by a Zilog Serial Communication Controller (SCC) chip on the BL1700. Channel A can be configured as two-wire RS-485 or five-wire RS-232. When configured as RS-232, Channel A also provides DCD and DTR signals. Synchronous communication is possible on this channel, but is not supported by Dynamic C drivers at this time. Channel A is not available on the BL1720 or BL1730.

Channel B

Channel B is a general-purpose serial channel. Along with Channel A, it is controlled by the Serial Communication Controller chip. Channel B can be configured as two-wire RS-485 or five-wire RS-232. Synchronous communication is possible on this channel, but is not supported by Dynamic C drivers at this time. Channel B is not available on the BL1720 or BL1730.

Operating Modes and Configuration

Table 3-10 and Table 3-11 show the operating modes and jumper configurations for the serial channels on the BL1700.

Table 3-10. Serial Channel Configuration Jumper Settings

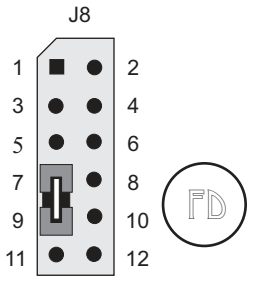
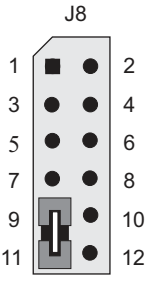
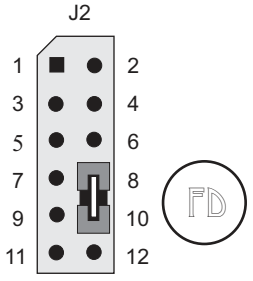
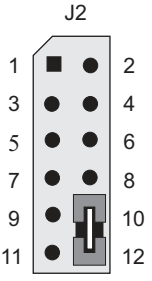
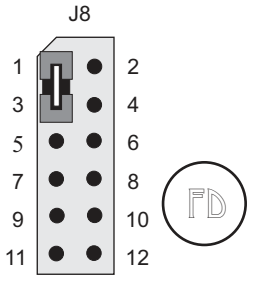
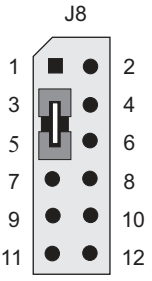
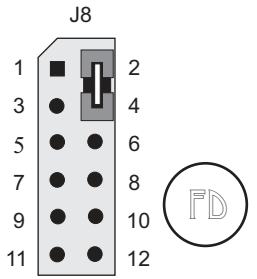
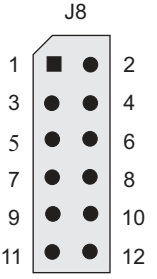
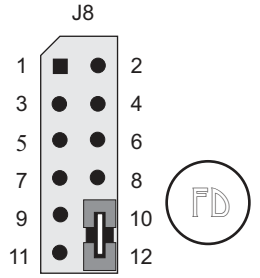
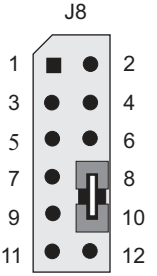
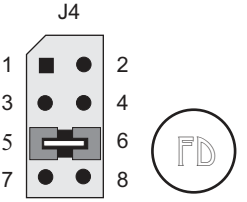
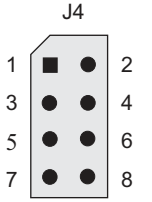
Channel	Jumper Settings	
	RS-232 Communication	RS-485 Communication
Channel 0	No jumper settings	
Channel 1	 3-wire RS-232	 2-wire RS-485
Channel A	 5-wire RS-232 +DCD +DTR	 2-wire RS-485
Channel B	 5-wire RS-232	 2-wire RS-485

Table 3-11. Serial Channel Configuration Jumper Settings

Channel	Jumper Settings	
	SCC Option	**User Application Option**
Channel A	 /DREQ0 used for SCC Channel A	 /DREQ0 available for user application
Channel B	 /DREQ1 used for SCC Channel B	 /DREQ1 available for user application
Channel A and Channel B	 /INT0 used for serial communication on Channel A and Channel B	 /INT0 available for user application

Configuring a Multidrop Network

- Configure the serial channels that you wish to use for RS-485 communication.
- On all networked controllers, connect RS-485+ to RS-485+ and RS-485- to RS-485- using single twisted pair wires (nonstranded, tinned).

Refer to the Dynamic C manuals for more details on master-slave networking.

RS-485 Termination

Termination and bias resistors are required in a multidrop network to minimize reflections (echoing), and to keep the network line active in an idle state. Typically, termination resistors are installed at the master node and the physical end node of an RS-485 network. Termination resistors are provided for Channel 1, Channel A, and Channel B configured as RS-485.

If you wish to configure a multidrop network, be sure to enable the 120 Ω termination resistors on both the master network controller and the "end" slave controller.

Figure 3-12 illustrates a multidrop network, and Table 3-12 provides the jumper settings to enable/disable the termination resistors.

External Connections

Each serial channel has its own individual header for external connections. Both RS-232 and RS-485 signal lines for Channel 1, Channel A, and Channel B are brought out to a serial channel's 10-pin header. Only one set of signals, RS-232 or RS-485, is active.

The three-wire RS-232 interface provides the following signals.

- RX
- TX
- GND

The five-wire RS-232 interface provides the following signals.

- RX
- TX
- RTS
- CTS
- GND

The two-wire RS-485 interface provides the following signals.

- RS-485+
- RS-485-

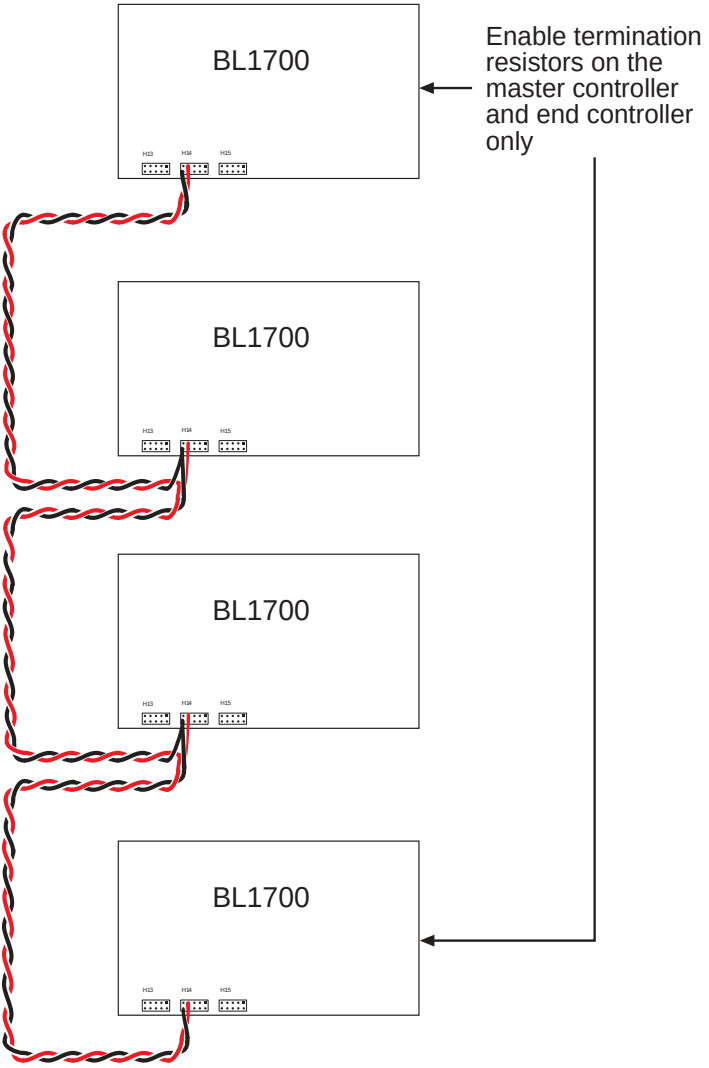
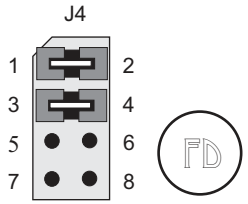
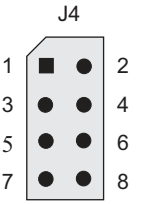
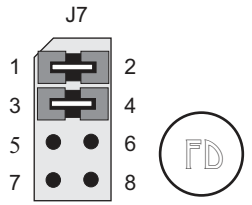
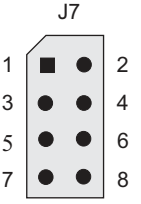
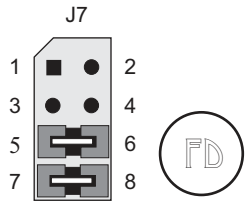
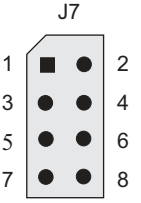


Figure 3-12. Multidrop Network

The RS-485 drivers supplied with the BL1700 support up to 32 nodes. The transmission bandwidth may be reduced as additional nodes over the benchmark quantity of 32 are added to the network. Contact Z-World Technical Support for assistance with large-scale network design.

Table 3-12. Termination Resistor Jumper Settings

Channel	Jumper Settings	
	Termination Resistors Enabled	**Termination Resistors Disabled**
Channel 0	No RS-485 available	
Channel 1		
Channel A		
Channel B		

Connections to the serial channels are made via the 10-pin headers shown in Figure 3-13. The headers are standard vertical 0.025" square (0.635 mm square) posts on 0.100" (2.54 mm) centers.

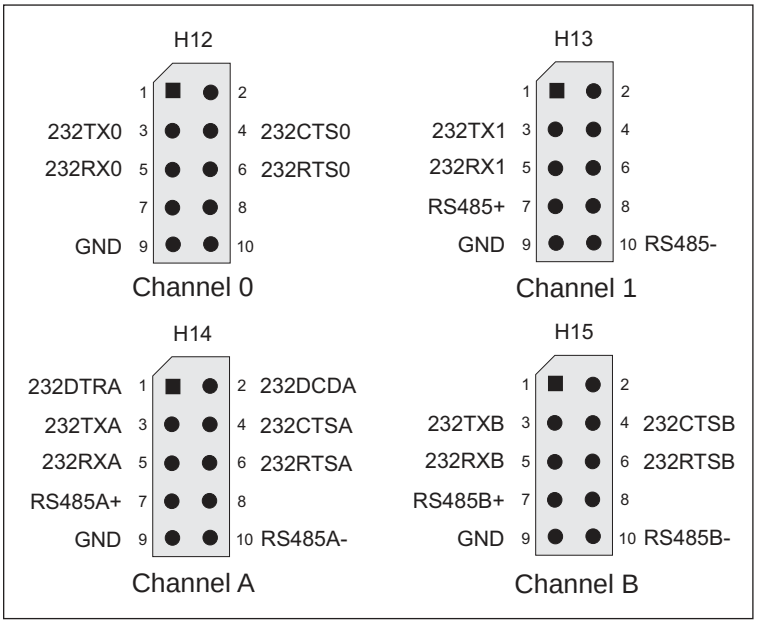


Figure 3-13. Pinouts of BL1700 Serial Communication Headers H12 through H15

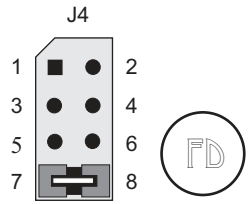
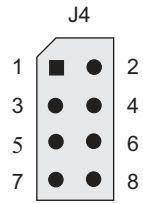
PLCBus

The PLCBus provides easy I/O expansion for the BL1700. PLCBus expansion boards provide additional I/O capacity, A/D converters, D/A converters, serial channels, relay outputs, stepper motor controllers, and more. Expansion boards are connected to the BL1700 via a 26-conductor ribbon cable. Several PLCBus expansion boards may be daisy-chained to increase the I/O capacity further. Dynamic C provides easy to use software for all Z-World expansion boards.

Operating Modes and Configuration

Some PLCBus expansion boards use the /AT line on the PLCBus. Jumpers on header J4 on the BL1700 determine whether the /INT1 signal is connected to the PLCBus /AT line, as shown in Table 3-13. If you intend to use a PLCBus expansion board that uses the /AT signal, make sure that a jumper is installed in the JP4:7-8 position. If you want to use the /INT1 signal for another external signal, and it is not needed for the PLCBus, then remove the jumper from the J4:7-8 position.

Table 3-13. BL1700 PLCBus Jumper Settings

/INT1 used as /AT on PLCBus	/INT1 external use only
	

External Connections

J5 is the PLCBus connector on the BL1700. PLCBus devices are connected with ribbon cables on 26-pin connectors.

Refer to Appendix E, “PLCBus,” for more detailed information on the PLCBus and Z-World’s expansion boards.

CHAPTER 4: SOFTWARE DEVELOPMENT

Chapter 4 describes how to use the features of the BL1700 Series controller. The following major sections are included.

- Supplied Software
- Digital Inputs
- Digital Outputs
- PWM Outputs
- Analog Inputs
- Serial Channels
- LED
- Additional Software

Supplied Software

Software drivers for controlling the BL1700's inputs/outputs are provided with Dynamic C. The library **EZIOBL17.LIB** provides drivers specific to the BL1700. In order to use **EZIOBL17.LIB** and other libraries, it is necessary to include the appropriate Dynamic C libraries. These libraries are listed in Table 4-1.

Table 4-1. BL1700 Software Libraries

Library	Application
AASC.LIB	All BL1700 serial communication applications
AASCURT2.LIB	XP8700 applications only
EZIOBL17.LIB	All BL1700 applications
EZIOPBDV.LIB	All expansion board applications
EZIOPLC2.LIB	All expansion board applications
STEP2.LIB	XP8800 applications only

Your application program can use these libraries by including them in your program. To include these libraries, use the **#use** directive as shown below.

```
#use eziobl17.lib
```

See the ***Dynamic C Technical Reference*** manual for more information on **#use** and other directives as well as other libraries.

Digital Inputs

The BL1700 is equipped with protected digital inputs designed as logical data inputs that return a 1 when the input is high or 0 when the input is low.

A low-pass filter on each input channel has a time constant of:

T_{RC} = 220 µs (4.5 kHz).

If the signals present on the digital inputs change states faster than this, the readings on the inputs may not be accurate.

How to Read the Input

This section provides information on using the Dynamic C software drivers for the BL1700's protected digital inputs.

The following software drivers read the status of the protected digital inputs.

- **`unsigned BankA( unsigned eioAddr )`**
- **`unsigned BankB( unsigned eioAddr )`**

  `BankA` converts **`eioAddr`** to a value of 16–31 for addressing the correct input or output assignments. **`BankB`** converts **`eioAddr`** to a value of 0–15.

  PARAMETER: **`eioAddr`** specifies channel number from 0–15.

  RETURN VALUE: the formatted I/O assignment, or –1 if the parameter **`eioAddr`** is out of range.

- **`int eioBrdDI( unsigned eioAddr )`**

  Reads the state from one of the 32 physical digital inputs. Sets **`eioErrorCode`** if **`eioAddr`** is out of range.

  PARAMETER: **`eioAddr`** specifies the input to be read. Valid numbers are from 0 to 31. 0–15 represents Bank B. 16–31 represents Bank A.

  RETURN VALUE: 0 if input reads low, 1 if input reads high.

- **`unsigned inport( unsigned port )`**

  Reads a value from an I/O port.

  PARAMETER 1: **`port`** is the BL1700 port address to read. When used to read the digital inputs, **`port`** is one of four groups of eight inputs. There are two groups of eight inputs for each bank.

  RETURN VALUE: The value read from the port.

Table 4-2 lists the addresses and corresponding headers of the digital input ports on the BL1700.

Table 4-2. Digital Input Addresses

Bank	Bank B		Bank A	
Header	H10	H7	H6	H9
Channels	HVB00–HVB07	HVB08–HVB15	HVA08–HVA15	HVA00–HVA07
Physical Channels	0–7	8–15	24–31	16–23
Address	0x4040	0x4041	0x4042	0x4043

The factory default is for Bank A to be configured for digital inputs.

The lower eight bits of the value read back by the inport function represent the status of the inputs. Bit 0 represents inputs 0, 8, 16, or 24, depending on which address is read. Bit 1 represents inputs 1, 9, 17, or 25, and so forth.

Sample Program

The sample program **`BL17DIO.C`** shows how to use the digital I/O. It can be found in the Dynamic C **`SAMPLES\BL17XX`** subdirectory.

Digital Outputs

The BL1700 provides up to 32 high-voltage, high-current driver outputs. Some outputs can also function as pulse width modulated (PWM) outputs. This section provides information on the Dynamic C software drivers for the BL1700's high-voltage driver outputs.

The following software function turns a specified high-voltage driver ON or OFF.

- **`unsigned BankA( unsigned eioAddr )`**
- **`unsigned BankB( unsigned eioAddr )`**

  `BankA` converts **`eioAddr`** to a value of 16–31 for addressing the correct input or output assignments. **`BankB`** converts **`eioAddr`** to a value of 0–15.

  PARAMETER: **`eioAddr`** specifies channel number from 0–15.

  RETURN VALUE: the formatted I/O assignment, or -1 if the parameter **`eioAddr`** is out of range.

- **`int eioBrdDO( unsigned eioAddr, char state )`**

  Sets the state of a digital output. Sets **`eioErrorCode`** if parameter **`eioAddr`** is out of range.

  PARAMETERS: **`eioAddr`** specifies the output to be set. Valid numbers are from 0 to 31. 0–15 represents Bank B. 16–31 represents Bank A.

  `state` is the desired output state for the specified output. A non-zero value turns the output on. A zero turns the output off.

  RETURN VALUE: Returns 0 if successful, -1 if **`eioAddr`** is out of range.

- **`void outport( unsigned port, unsigned value )`**

  Writes data to an I/O port.

  PARAMETERS: **`port`** is the BL1700 port address to be written. When used to write to the digital outputs, **`port`** is one of four groups of eight outputs. There are two groups of eight outputs for each bank.

  `value` is the data to be written to the port. When used to write to the digital outputs, data bits D3, D2, and D1 determine which output in a group is selected. Data bit D0 determines the state of the output. Data bits D7 through D4 are unused.

Table 4-3 shows the address and data values used with the **`outport`** function for writing to the digital outputs.

Table 4-3. Digital Output Addresses

Bank B HVB00–HVB15		Address	OFF data	ON data	Bank A HVA00–HVA15		Address	OFF data	ON data
H10	0	0x4100	0	1	H9	16	0x4110	0	1
	1	0x4100	2	3		17	0x4110	2	3
	2	0x4100	4	5		18	0x4110	4	5
	3	0x4100	6	7		19	0x4110	6	7
	4	0x4100	8	9		20	0x4110	8	9
	5	0x4100	10	11		21	0x4110	10	11
	6	0x4100	12	13		22	0x4110	12	13
	7	0x4100	14	15		23	0x4110	14	15
H7	8	0x4108	0	1	H6	24	0x4118	0	1
	9	0x4108	2	3		25	0x4118	2	3
	10	0x4108	4	5		26	0x4118	4	5
	11	0x4108	6	7		27	0x4118	6	7
	12	0x4108	8	9		28	0x4118	8	9
	13	0x4108	10	11		29	0x4118	10	11
	14	0x4108	12	13		30	0x4118	12	13
	15	0x4108	14	15		31	0x4118	14	15

The factory default is for Bank B to be configured for digital outputs.

Sample Program

The sample program **`BL17DIO.C`** shows how to use the digital I/O. It can be found in the Dynamic C **`SAMPLES\BL17XX`** subdirectory.

Pulse-Width Modulated (PWM) Outputs

Digital outputs 0–6 on Bank B can produce fixed-frequency, pulse-width modulated (PWM) signals. When these outputs are being used for PWM operation, Channel 7 is used by software to support PWM and cannot be used for your application.

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). Using the supplied software, generating PWM signals consumes about 8% of controller's processing power.

When PWM functions are used, serial communication baud rates may be affected because of an overloading of the microprocessor's resources. In addition, serial data rates become limited and fixed at 4800 bps for Serial Port 1. Be sure to reset the Dynamic C baud rates to 4800 bps.

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

How to Use the PWM Feature

The BL1700 can produce fixed-frequency, fixed-phase, variable-duty-cycle square waves from up to seven of its outputs. Figure 4-1 and Figure 4-2 show PWM transition and DMA timing.

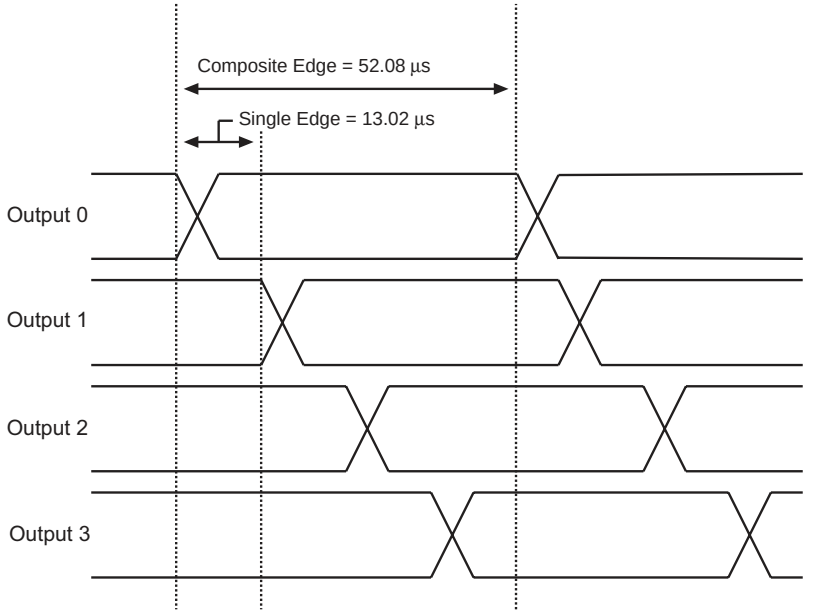


Figure 4-1. Transition Timing

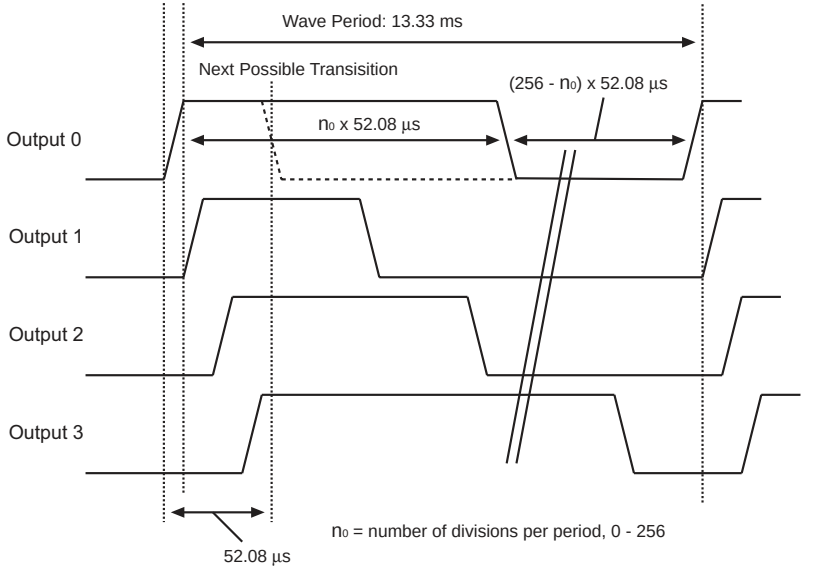


Figure 4-2. DMA Timing

Notice that each square wave's period is exactly 1024 "divisions." One division equals 120 clock cycles (120/9.216 MHz = 13.02 µs) for the PWM function. Consequently, the period of each square wave is 1024 × 13.02 µs = 13.33 ms.

Notice also that the square waves are displaced slightly from each other in phase. That is, output 1s output starts and ends one division after output 0s, output 2s one division after output 1s, and output 3s one division after output 2s As a result, although the period of each wave is 1024 divisions, a change to one particular channel is possibly only every 4 divisions. Therefore, the resolution of the transition edge in the wave is 1/256.

PWM Software

The supplied software provides two levels of support. The first level provides easy-to-use fixed PWM functions for only four of the outputs (outputs 0–3). The periods of the PWM signals are fixed at 13.3 ms (75 Hz), with a resolution of 256 division per period (8-bit resolution). Using the supplied software, generating PWM signals consumes about 8% of the controller's processing power. The second PWM support level allows you to create custom PWM functions for seven of the outputs (outputs 0–6).

The following three functions are the first level functions. They are designed for ease of use. These functions are located in **`EZIODPWM.LIB`** that is automatically included when **`EZIOBL17.LIB`** is included.

- **`int eioBrdAO( unsigned eioAddr, unsigned state )`**

  Specifies the duty cycle for a particular output channel. Set **`eioErrorCode`** if **`eioAddr`** is out of range.

  PARAMETERS: **`eioAddr`** is a number ranging from 0 to 3.

  `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 (e.g., 128 for 50% duty cycle, 64 for 25% duty cycle).

  RETURN VALUE: 0 if successful, –1 if not.

The PWM functions use the Z180's built-in DMA hardware. The use of DMA-driven PWM limits the communication speed of the Z180's Serial Port 1 to 4800 bps. In addition, the Z180 effectively runs at least 8% more slowly.

Be sure your application calls **_eioBrdAORf** at least every 25 ms to refresh the drivers' period.

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

- **void _eioSetupAO1st ()**

  Initializes the PWM hardware.

  _eioSetupAO1st must be called before using **eioBrdAO**.

- **int _eioBrdAORf ()**

  Refreshes the DMA counter and address pointer.

  Your program must call it every 25 ms (or more frequently) after **_eioSetupAO1st** is called.

  RETURN VALUE: The function returns -1 if the DMA count is zero (PWM has stopped), and returns 0 otherwise. If the function returns -1, the driver is either not initialized (by calling **_eioSetupAO1st**), or **_eioBrdAORf** is not called at least every 25 ms.

Sample Program

BL17PWM4.C is a sample program that shows how to use the pulse width modulation feature using the functions listed above. It can be found in the Dynamic C directory under **SAMPLES\BL17XX**.

Analog Inputs

The BL1700's analog inputs provide an easy-to-use interface to a wide variety of sensors and transducers. The BL1700 provides 10 single-ended A/D conversion channels with 12-bit resolution.

Using the Analog Inputs

The factory calibrates each BL1700, storing each unit's individual zero offset and actual gain for its eight primary channels in simulated EEPROM. Your application can use library functions to access the simulated EEPROM's calibration constants to correct measurements for offset and gain error.

- **`void eioBrdInit( int flags )`**

  Initializes the analog-to-digital converter to the default output mode. The default mode is unipolar input, 12-bit data length, most significant bit first.

  PARAMETER: **`flags`** is not used at this level and should be set to 0.

  Call **`eioBrdInit`** before calling **`eioBrdAI`**.

- **`int eioBrdAI( unsigned eioAddr )`**

  Reads one of the 10 voltage inputs and performs analog-to-digital conversion. Sets **`eioErrorCode`** if **`eioAddr`** is out of range.

  PARAMETER: **`eioAddr`** specifies an input number of 0 to 9 or 16 to 25 to be read. **`eioAddr`** values 0 through 9 represent analog inputs 0 through 9, and will cause the function to return the voltage read on an input. **`eioAddr`** values 16 through 25 also represent analog inputs 0 through 9, but cause the function to return a 12-bit raw data value for the analog input.

  RETURN VALUE: The function returns the voltage read as a real number in a floating-point representation for **`eioAddr`** values 0–9 if the read is successful. For **`eioAddr`** values 16–25, if the read is successful, the function returns a floating-point representation of an unsigned integer value (0–4095) for the 12-bit raw data value read from the A/D converter.

- **`int eioBrdAdcMode( int datalen, int dataformat, int polarformat )`**

  Sets the analog-to-digital conversion data length, data format, and polarity format other than default. Call this function after eioBrdInit and before eioBrdAI.

  RETURN VALUE: returns 1 if successful, -1 if an invalid parameter is passed to the function.

Call **`eioBrdAdcMode`** after calling **`eioBrdInit`** and before calling **`eioBrdAI`**.

Table 4-4 shows the parameters **`datalen`**, **`dataformat`**, and **`polarformat`**.

Table 4-4. Analog-to-Digital Converter Modes

Parameter	Value
`datalen`	0 – 12-bit data length 1 – 8-bit data length 2 – 12-bit data length 3 – 16-bit data length
`dataformat`	0 – most significant bit first 1 – least significant bit first
`polarformat`	0 – unipolar 1 – bipolar

- **`int eioBrdACalib( int eioAddr, unsigned d1, unsigned d2, float v1, float v2 )`**

  Calculates the calibration constants for an analog input channel using two known voltages and two corresponding raw data readings. Stores the calibration constants in EEPROM.

  PARAMETERS: **`eioAddr`** is the analog input channel.

  `d1` is the raw data corresponding to **`v1`**.

  `d2` is the raw data corresponding to **`v2`**.

  `v1` is the known voltage used to obtain **`d1`**.

  `v2` is the known voltage used to obtain **`d2`**.

  RETURN VALUE: 0 if successful, –1 if **eioAddr** is out of range.

Since the BL1700 is calibrated at the factory, it is only necessary to use this function to recalibrate the BL1700.

Sample Program

`BL17AIN.C` is a sample program that shows how to use the analog inputs. It can be found in the Dynamic C directory under **`SAMPLES\BL17XX`**.

Serial Channels

The BL1700 and BL1710 provide four serial communication channels. Three of the ports can be configured as RS-232 or RS-485. This section provides information on RS-232 and RS-485 communications.

The BL1720 and BL1730 have only two serial communication channels. One of these channels is a dedicated RS-232 channel, the other is configurable as either RS-232 or RS-485.

Chapter 3, "BL1700 Hardware," provides information on configuring the serial channels.

RS-232 Communication

The RS-232 channels and the supplied Dynamic C software allows the BL1700 to communicate with other computers or controllers. By adding a modem, remote communications can be achieved (including remote downloading) using the X-modem protocol. Examples of RS-232 software drivers can be found in the Dynamic C **\SAMPLES\AASC** directory.

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

Use the optional Z-World SIB2 if you need to make all of the serial channels available to your application during software development.

See Chapter 2, "Getting Started," and Appendix D, "Serial Interface Board 2," for more information.

RS-485 Communication

The BL1700 can be configured to provide up to three channels of RS-485 communications. RS-485 is an asynchronous multi-drop half-duplex standard that provides multi-drop networking with maximum cable lengths up to 4000 feet.

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

This RS-485 hardware standard supports up to 32 controllers on one network. The supplied software supports 1 master unit, plus up to 255 slave units (which may consist of any combination of Z-World controllers that support the RS-485 protocol).

Software

Serial channels 0 and 1 are available on all versions of the BL1700. These serial channels are supported by Dynamic C library functions.

Serial channels A and B are driven by U13 (a Zilog Serial Communication Controller). The BL1720 and BL1730 do not have this chip installed, therefore, channels A and B are not supported on the BL1720 or BL1730. Serial channels A and B have additional capabilities beyond those supported by the Dynamic C libraries. If you would like to use these additional capabilities, refer to the Zilog ***Serial Communication Controllers Manual***.

Comprehensive information on the serial channel software and programming can be found in the ***Dynamic C Function Reference*** manual and the ***Dynamic C Application Frameworks*** manual.

The following functions are used with the RS-485 serial channels on the BL1700.

- **`int sccSw485( unsigned channel, unsigned state )`**

  Enables or disables the RS-485 drivers for Channel A or Channel B on the SCC.

  PARAMETERS: **`channel`** is **`SCC_A`** or **`SCC_B`**.

  `state` is 1 to enable the driver, 0 to disable it.

  RETURN VALUE: 0 if **`channel`** is valid, -1 if not.

- **`int z1Sw485( unsigned state )`**

  Enables or disables the RS-485 driver for Channel 1.

  PARAMETER: **`state`** is 1 to enable the driver, 0 to disable it.

  RETURN VALUE: 0 if **`channel`** is valid.

Sample Program

`BL17SCC.C` is a serial communication sample program found in the Dynamic C **`SAMPLES\BL17XX`** directory.

LED

LED D2 is a general-purpose device that can be turned on and off under software control by using this function.

- **`int switchLED( unsigned state )`**

  Turns LED D2 on or off.

  PARAMETER: **`state`** is 1 to turn the LED on, 0 to turn it off.

  RETURN VALUE: 0 if **`state`** is valid, -1 otherwise.

Additional Software

- For real-time clock information, refer to descriptions of functions **`tm_rd`** and **`tm_wr`** in your Dynamic C manuals.
- For watchdog information, refer to descriptions of the function **`hitwd`** in your Dynamic C manuals.
- For simulated EEPROM information, refer to descriptions of the functions **`ee_rd`** and **`ee_wr`** in Appendix G.
- For power failure flag information, refer to the descriptions of the function **`_sysIsPwrFail`** and **`sysIsPwrFail`** in your Dynamic C manuals.
- For resetting the board information, refer to descriptions of the functions **`sysForceSupRst`**, **`sysIsSuperReset`**, **`_sysIsSuperReset`**, **`sysForceReset`**, **`_sysIsWDTO`**, and **`sysIsWDTO`** in your Dynamic C manuals.

APPENDIX A: TROUBLESHOOTING

Appendix A provides procedures for troubleshooting system hardware and software. The following sections are included.

- Out of the Box
- Dynamic C Will Not Start
- Finding the Correct COM Port and Baud Rate
- BL1700 Resets Repeatedly
- Troubleshooting Software

Out of the Box

Check the items listed below before starting development. Rechecking may help to solve problems found during development.

- Do not connect any boards with PLCBus, RS-485 or any other I/O devices until you verify that the BL1700 runs standalone.
- Verify that your entire system has a good, low-impedance ground. The BL1700 is often connected between the PC and some other device. Any differences in ground potential from unit to unit can cause serious, hard-to-diagnose problems.
- Double-check the connecting cables.
- Verify that your PC's COM port actually works. Try connecting a known-good serial device to your COM port. Remember that on a PC COM1/COM3 and COM2/COM4 share interrupts. User shells and mouse software, particularly, often interfere with proper COM-port operation. For example, a mouse running on COM1 can preclude your running Dynamic C on COM3, unless the interrupt is changed.
- Use the supplied Z-World power supply. If you must use your own power supply, verify that it has enough capacity to support the BL1700 and is adequately filtered.
- Use the supplied Z-World cables. The most common fault of home-made cables is their failure to properly assert CTS at the RS-232 port of the BL1700. Without CTS's being asserted, the BL1700's RS-232 port will not transmit. You can assert CTS by either connecting the RTS signal of the PC's COM port or looping back the BL1700's RTS.
- Experiment with each peripheral device you connect to your BL1700 to determine how it appears to the BL1700 when it is powered up, powered down, when its connecting wiring is open, and when its connecting wiring is shorted.

LCD Connected to BL1700 Does Not Work

Under extreme conditions, some LCDs connected to a BL1700 via the PLCBus may fail to function. The main reason for this is that the 18.432 MHz clock speed of the BL1700 is too fast for the LCD connected via the PLCBus port. The easiest software solution is to add a line to the application to slow down the clock speed, but this will impact other functions such as the serial rate and the PRT timer that depend on the clock speed. If a 9.216 MHz clock speed is adequate, then the BL1700 is available with a CM7210 core module, which features a 9.216 MHz clock.

The BL1730 comes with a CM7210 core module, but only has two serial ports.

Dynamic C Will Not Start

If Dynamic C will not start, an error message on the Dynamic C screen (for example, **Target Not Responding** or **Communication Error**), announces a communication failure.

You could have one or more of the following problems in series:

- You have selected the wrong COM port.
- You need to reset the BL1700 (press reset switch SW1).
- You have not connected the wiring properly.

The first thing to check is the hardware and software setup of your PC's COM port. Areas to check are listed below.

- Ensure that all wiring and cables are connected properly.
- Ensure that you have selected the proper COM port.

Most PCs have at least two COM ports (COM1 and COM2), while some computers have additional COM ports. Sometimes a PC assigns COM1 or COM2 to an internal modem, leaving the other COM port available on the back of the PC.

Some PCs have special programs to reconfigure their port assignments. You may need to run such a program to make a given COM port appear at an external back panel "D" connector.

Repeat the following procedure until you find a COM port that works with Dynamic C and your BL1700.

1. Use the Serial command of Dynamic C's **Options** menu to try a different COM port.
2. Reset the BL1700 by pressing reset switch SW1.
3. Select **Reset Target** from the **Run** menu. Dynamic C tries to establish communication again.

BL1700 Resets Repeatedly

If the program fails to hit the watchdog timer periodically, the watchdog timer causes a reset every 1.0 second. When you debug a program using the Dynamic C debugger, Dynamic C hits the watchdog timer. If your program does not hit the watchdog timer , then you will have trouble running your program in standalone mode. (To hit the watchdog, make a call to the Dynamic C library function **hitwd**).

- **Dynamic C looses link with application program**

If your program disables interrupts for a more than 50 ms, Dynamic C may lose its link with the BL1700.

Troubleshooting Software

Symptom: The DMA-driven PWM correctly drives the output for a while, then suddenly some channels remain ON, others remain off.

Cause: Most likely, the function **_eioBrdAORf()** is not called frequently enough.

·**Resolution:** There are three possible solutions. One is to increase the frequency of calling **_eioBrdAORf()**, the other is to increase the size of the waveform pattern buffer. The third solution is to slow down the clock **CKA1**.

Refer to the section “PWM Addressing Detail” in Appendix F, “Advanced Programming,” for more details.

APPENDIX B: SPECIFICATIONS

Appendix B provides comprehensive BL1700 physical, electronic and environmental specifications.

Electronic and Mechanical Specifications

Table B-1 lists the electronic, mechanical, and environmental specifications for the BL1700.

Table B-1. BL1700 General Specifications

Parameter	Specification
Board Size	4.20″ × 6.25″ × 0.85″ (107 mm × 159 mm × 21.6 mm)
Operating Temperature	–40°C to 70°C
Humidity	5% to 95%, noncondensing
Power	15 V DC to 30 V DC, 140 mA
Digital Inputs	16 standard, up to 32 possible at expense of outputs
Digital Outputs	16 standard, up to 32 possible at expense of inputs
Analog Inputs	Ten 12-bit channels: • 8 conditioned, factory configured 0 V to 10 V • 2 unconditioned, 0 V to 2.5 V
Analog Outputs	Pulse-width modulated, on digital output lines
Resistance Measurement Input	No
Processor	Z180
Clock	18.432 MHz standard
SRAM	32K standard, supports up to 512K
Flash EPROM	128K standard, supports up to 256K, up to 512K EPROM possible
Serial Ports	• 1 full-duplex RS-232 • 3 configurable as full-duplex RS-232 or as RS-485
Serial Rate	Up to 57,600 bps
Watchdog	Yes
Time/Date Clock	Yes
Backup Battery	Panasonic BR2325-1HG 3 V DC lithium ion, rated life 190 mA·h

BL1700 Mechanical Dimensions

Figure B-1 shows the mechanical dimensions for the BL1700.

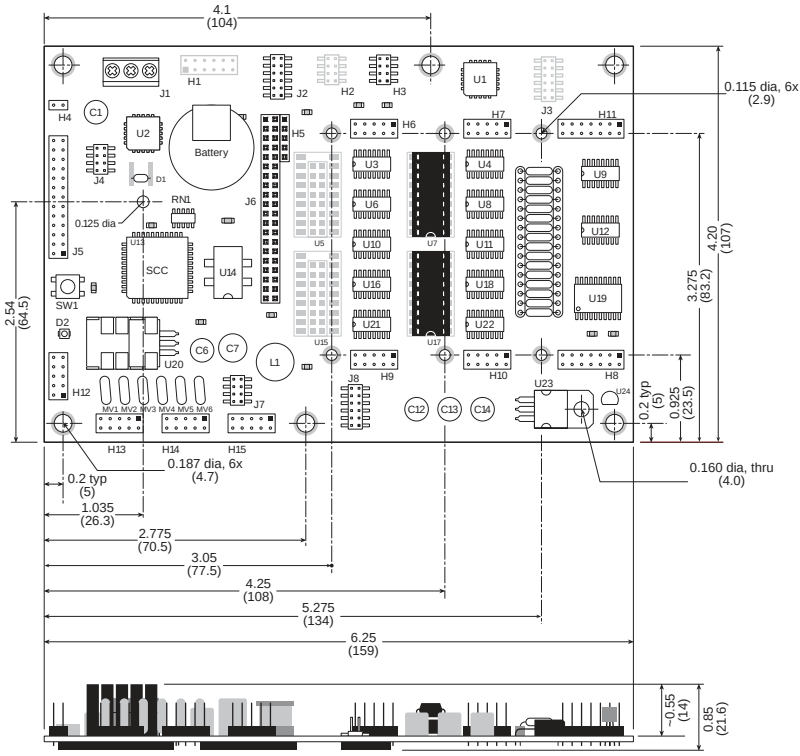


Figure B-1. BL1700 Dimensions

The dimensions shown above include the CM7200 that is part of the BL1700. With the tallest field-wiring terminal (FWT-Opto) attached, the height of the BL1700 assembly becomes 1.35" (34.3 mm).

Header and Jumper Information

Table B-2 lists the header functions for the input/output and serial communication headers. The header locations are shown in Figure B-2.

Table B-2. BL1700 Header Functions

Header	Function
H1	Serial communication and interrupts (optional)
H6	Digital input/output
H7	Digital input/output
H8	Analog input
H9	Digital input/output
H10	Digital input/output
H11	Analog input
H12	Channel 0 RS-232 serial communication port
H13	Channel 1 RS-232/RS-485 serial communication port
H14	Channel A RS-232/RS-485 serial communication port
H15	Channel B RS-232/RS-485 serial communication port
J1	Power input
J5	PLCBus

Table B-3 provides the relevant pin 1 locations for these headers.

Table B-3. BL1700 Pin 1 Locations (in inches)

Header	Location
H1	1.5, 3.95
H6	3.7, 3.375
H7	4.9, 3.375
H8	6.1, 3.375
H9	3.7, 0.925
H10	4.9, 0.925
H11	6.1, 0.925
H12	0.2, 0.5
H13	1.0, 0.25
H14	1.7, 0.25
H15	2.4, 0.25
J5	0.2, 2.015

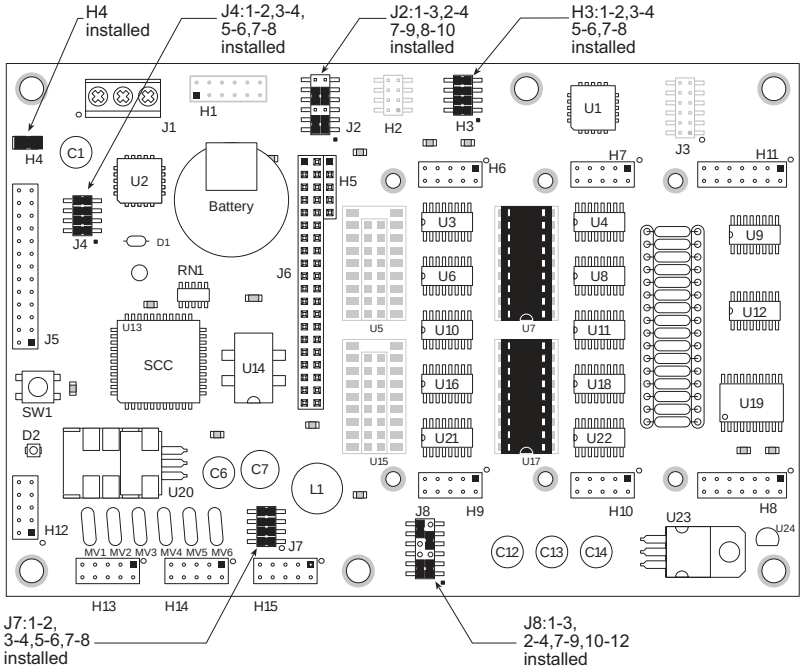


Figure B-2. BL1700 Headers and Factory Default Jumper Configurations

Table B-4 lists the jumper configurations for the BL1700 configurable headers.

Table B-4. Standard BL1700 Jumper Settings

Header	Pins	Description	Factory Default
H3	1–2 3–4	Connect for HVB00–HVB07 sinking output	Connected
	1–3 4–4	Connect for HVB00–HVB07 sourcing output	
	5–6 7–8	Connect for HVB08–HVB15 sinking output	Connected
	5–7 6–8	Connect for HVB08–HVB15 sourcing output	
H4	1–2	Connected for Program Mode, disconnected for Run Mode	Connected

continued…

Table B-4. Standard BL1700 Jumper Settings (continued)

Header	Pins	Description	Factory Default
J2	1–3	Connect for HVA08–HVA11 inputs pulled up	Connected
	3–5	Connect for HVA08–HVA11 inputs pulled down	
	2–4	Connect for HVA12–HVA15 inputs pulled up	Connected
	4–6	Connect for HVA12–HVA15 inputs pulled down	
	7–9	Connect for HVA00–HVA07 inputs pulled up	Connected
	9–11	Connect for HVA00–HVA07 inputs pulled down	
	8–10	Connect for 5-wire RS-232, DCD and DTR	Connected
	10–12	Connect for 2-wire RS-232	
J4	1–2 3–4	Connect to enable RS-485 termination resistors for Channel 1	Connected
	5–6	Connect for /INT0 serial communication on Channel A and Channel B, disconnecto toallow /INT0 for user application	Connected
	7–8	Connect to allow /INT1 to be used as /AT on PLCBus, disconnect for /INT1 external use only	Connected
J7	1–2 3–4	Connect to enable RS-485 termination resistors for Channel A	Connected
	5–6 7–8	Connect to enable RS-485 termination resistors for Channel B	Connected

continued…

Table B-4. Standard BL1700 Jumper Settings (concluded)

Header	Pins	Description	Factory Default
J8	1–3	Connect to enable 5-wire RS-232 on Channel B	Connected
	3–5	Connect to enable 2-wire RS-485 for Channel B	Connected
	2–4	Connect to allow /DREQ0 to be used for Channel A	Connected
	4–6	Connect to allow /DREQ0 to be used for PWM, disconnect for user application	
	2, 4, 6	Disconnected, /DREQ0 available for user application	
	7–9	Connect for 3-wire RS-232 on Channel 1	Connected
	9–11	Connect for 2-wire RS-485 on Channel 1	
	10–12	Connect to allow /DREQ1 to be used for Channel B	Connected
	8–10	Connect to allow user application for Channel B	

Table B-5 lists the jumper settings for optional BL1700 configurations. These optional configurations involve adding or removing input interface or high-voltage driver ICs, which are surface-mounted. This work is most easily done in the factory in response to customer needs.

Table B-5. BL1700 Jumper Settings for Optional Inputs/Outputs

Header	Pins	Description
Bank A Digital Outputs		
H2	1–2 3–4	Connect for HVA00–HVA07 sinking output
	1–3 4–4	Connect for HVA00–HVA07 sourcing output
	5–6 7–8	Connect for HVA08–HVA15 sinking output
	5–7 6–8	Connect for HVA08–HVA15 sourcing output
Bank B Digital Inputs		
J3	1–3	Connect for HVB0–HVB03 inputs pulled up
	3–5	Connect for HVB00–HVB03 inputs pulled down
	2–4	Connect for HVB04–HVB07 inputs pulled up
	4–6	Connect for HVB04–HVB07 inputs pulled down
	7–9	Connect for HVB08–HVB15 inputs pulled up
	9–11	Connect for HVB08–HVB15 inputs pulled down

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

Protected Digital Inputs

Table B-6 lists the specifications for the protected digital inputs.

Table B-6. BL1700 Protected Digital Input Specifications

Protected Digital Inputs	Absolute Maximum Rating
Input Voltage	-20 V DC to +24 V DC, protected against spikes to ±48 V
Logic Threshold	2.5 V
Input Current	–15 mA to +15 mA
Leakage Current	5 µA
Noise/Spike Filter	Low-pass filter, RC time constant 220 µs
Frequency Response (worst case)	• Faster than 656 Hz • Not slower than 1.52 ms (input at 5 V DC)

Frequency Response for the Protected Inputs

The protection network comprises a low-pass filter with a corner frequency of 724 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 BL1700's data bus.

Equation (B-1) shows how R_{IN} and C affect the frequency response of the protected inputs HVA00 through HVA15.

$$f_c = [2\pi R_{IN} C]^{-1} = [(2\pi)(22 \times 10^3)(10^{-8})]^{-1} \quad \text{(B-1)}$$

$$f_c = 724 \text{ Hz}$$

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

Figure B-3 shows the protected input circuitry for protected inputs HVA00 to HVA15 in the factory default pulled-up configuration.

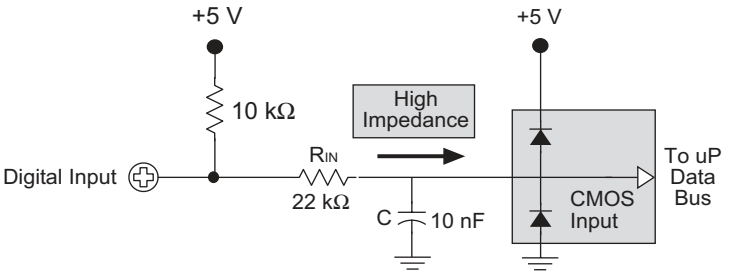


Figure B-3. Protected Input Circuitry, HVA00 through HVA15

If a faster frequency response is needed, it is possible to replace R_{IN} with a smaller value. For example, if the digital input is being driven by a +5 V DC CMOS compatible driver, R_{IN} can be replaced with a zero-ohm 0805 resistor.

> Replacing R_{IN} with a zero-ohm resistor will adversely affect the BL1700's noise immunity.

High-Voltage Drivers

Table B-7 lists the high-voltage driver characteristics when sinking drivers or sourcing drivers are used.

Table B-7. High-Voltage Driver Characteristics

Characteristic	Sinking Driver	Sourcing Driver
IC	2803	2985
Number of Channels	8	8
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	3 V to 30 V DC
Package Power Dissipation	2.2 W	2.2 W
Max. Current (all channels ON)	1.38 A	1.38 A
Max. Collector-Emitter Voltage (VCE)	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

For additional information on maximum operating conditions for the BL1700 high-voltage drivers, call Z-World Technical Support at (530) 757-3737.

Sinking Driver

The sinking-driver IC can handle a maximum of 1.38 A (500 mA for any channel), or 75 mA per channel on average if all channels are ON, at 60°C. The absolute maximum power that the driver IC can dissipate depends on several factors. The sinking IC's saturation voltage is 1.6 V DC max per channel.

The sinking driver's source voltage must range from 2 V to 48 V DC.

Sourcing Driver

The sourcing-driver IC can handle a maximum of 1.38 A (250 mA for any channel), or 75 mA per channel on average if all channels are ON, at 60°C. The sourcing IC can dissipate a maximum of 2.2 W. The saturation voltage is 1.6 V DC max per channel.

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.

For more information on sinking and sourcing high-voltage drivers, refer to the Motorola (DL128) or Allegro (AMS 502Z) linear data books.

See Appendix D, "Sinking and Sourcing Drivers," for more information on installing and using sourcing drivers.

APPENDIX C: ***FIELD WIRING TERMINALS (FWT) AND DIN RAILS***

Field Wiring Terminals

Discrete input/output lines may be connected to headers on the BL1700 Series of controllers with field wiring terminal (FWT) modules. This eliminates the need for ribbon cables. The optional quick-disconnect modules provide screw terminals for simple wiring.

The FWT38, FWT50, and FWT-Opto modules mate to two of the BL1700 Series board headers (H6–H9 and H7–H10) in any combination. This is equivalent to 20 connections per module.The FWT-A/D module mates with headers H8–H11 only in one position.

Figure C-1 illustrates the mounting configuration for the FWT modules and the CM7200.

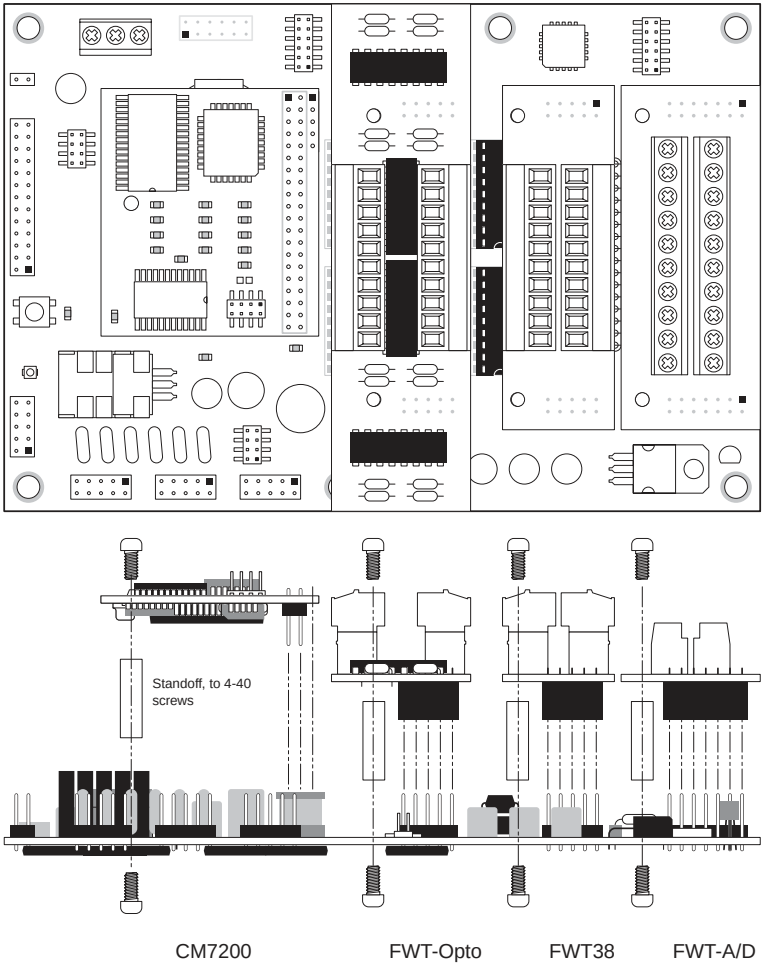


Figure C-1. BL1700 FWT and CM7200 Installation

These four FWT styles described in this section are available from Z-World. Your application may use a different arrangement than that shown in Figure C-1.

FWT38

The FWT38 has 20 terminals in two groups with 10 terminals each. Each group of terminals may be removed independently.

Table C-1 summarizes the specifications for the FWT38.

Table C-1. FWT38 Specifications

Parameter	Specification
Total I/O Channels	16
Screw Terminal Pitch	3.81 mm
Maximum Wire Gauge	28-16 AWG
Quick-Disconnect Capability	Wiring banks can be unplugged from the board separately (Phoenix Combicon type connection)
Wire Orientation	Top-exiting wires

Figure C-2 provides the dimensions for the FWT38.

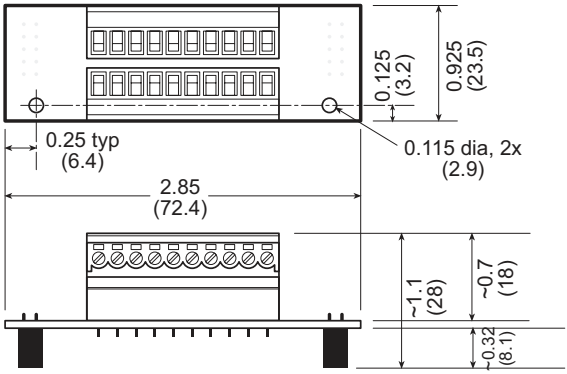


Figure C-2. FWT38 Dimensions

Figure C-3 shows the I/O channel assignments and pinouts for the FWT38.

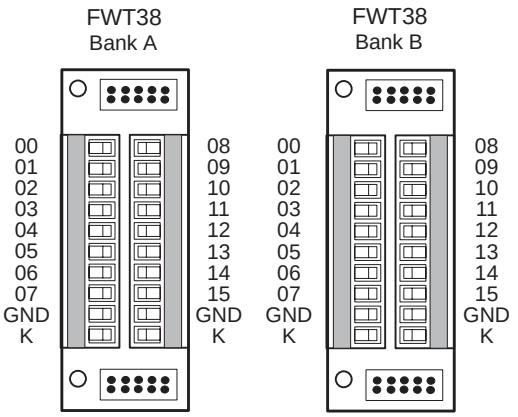


Figure C-3. FWT38 Pinouts

FWT50

The FWT50 provides 20 screw terminals. The terminal connectors are fixed to the FWT module and cannot be removed.

Table C-2 summarizes the specifications for the FWT50.

Table C-2. FWT50 Specifications

Parameter	Specification
Total I/O Channels	16
Screw Terminal Pitch	5.00 mm
Maximum Wire Gauge	24-12 AWG
Quick-Disconnect Capability	Unplugs from the BL1700 board as a single unit
Wire Orientation	Side-exiting wires

Figure C-4 provides the dimensions for the FWT50.

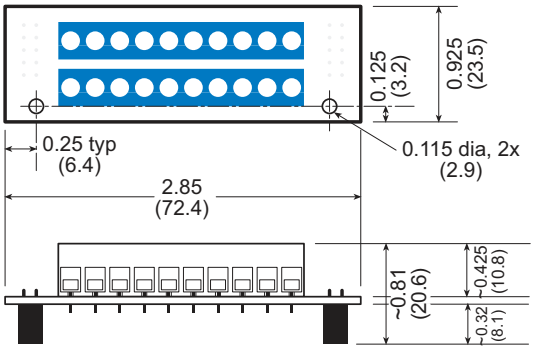


Figure C-4. FWT50 Dimensions

Figure C-5 shows the I/O channel assignments and pinouts for the FWT50.

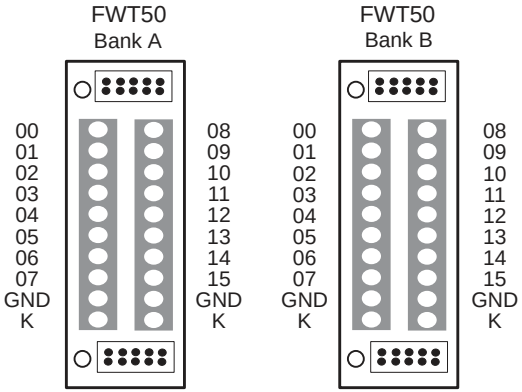


Figure C-5. FWT50 Pinouts

FWT-Opto

The FWT-Opto provides optical isolation to the input channels. The FWT-Opto is used only for inputs, and is not used if the BL1700 banks are all configured as outputs. All 16 channels must be committed to inputs when an FWT-Opto module is used.

Every four FWT-Opto inputs share a common return. The excitation resistors need to be pulled up to +5 V when the FWT-Opto module is used.

Table C-3 summarizes the specifications for the FWT-Opto.

Table C-3. FWT-Opto Specifications

Parameter	Specification
Total Input Channels	16 optically isolated input channels only
Screw Terminal Pitch	3.81 mm
Maximum Wire Gauge	28-16 AWG
Quick-Disconnect Capability	Wiring banks can be unplugged from the board separately (Phoenix Combicon type connection)
Wire Orientation	Top-exiting wires
Input Protection Range	5 kV rms between input and output
Maximum Input Voltage	±40 V
Guaranteed Input Switching Threshold	±9.5 V

The FWT-Opto module uses 4.7 kΩ input resistors to accommodate the large range of input voltages. This limits the input switching threshold to ±9.5 V. These 4.7 kΩ input resistors need to be replaced with 1.2 kΩ input resistors to handle smaller input voltages such as 5 V logic. If 0.125 W resistors are used, this will limit the maximum input voltage to ±12.2 V.

Figure C-6 provides the dimensions for the FWT-Opto module.

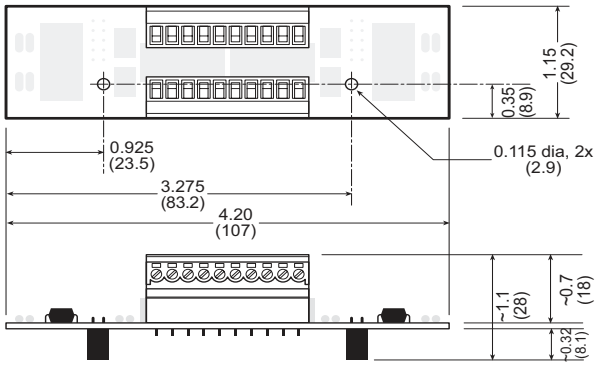


Figure C-6. FWT-Opto Dimensions

Figure C-7 shows the input channel assignments and pinouts for the FWT-Opto module.

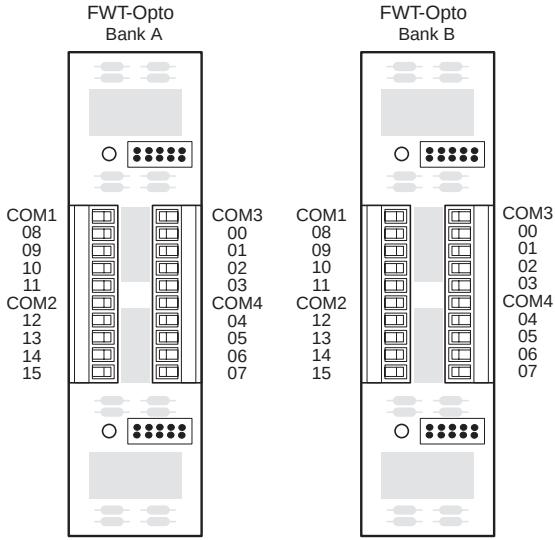


Figure C-7. FWT-Opto Pinouts

Figure C-8 shows an FWT-Opto optical isolation circuit.

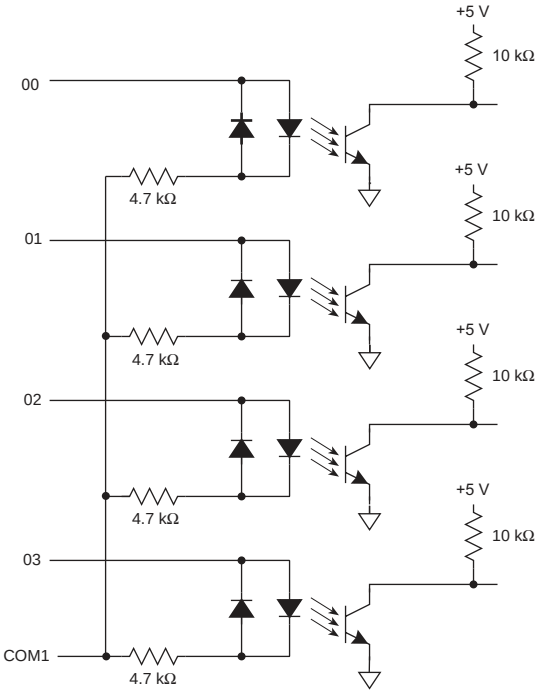


Figure C-8. FWT-Opto Optical Isolation Circuit

The opto-isolated inputs share a common return in groups of four. The software channel assignments remain the same for Banks A and B.

FWT-A/D

The FWT-A/D provides 20 screw terminals. The terminal connectors are fixed to the FWT module and cannot be removed.

The FWT-A/D is used only to access the analog inputs on the BL1700.

Table C-4 summarizes the specifications for the FWT-A/D.

Table C-4. FWT-A/D Specifications

Parameter	Specification
Total Input Channels	10
Screw Terminal Pitch	5.00 mm
Maximum Wire Gauge	24-12 AWG
Quick-Disconnect Capability	Unplugs from the BL1700 board as a single unit
Wire Orientation	Side-exiting wires

Figure C-9 provides the dimensions for the FWT-A/D.

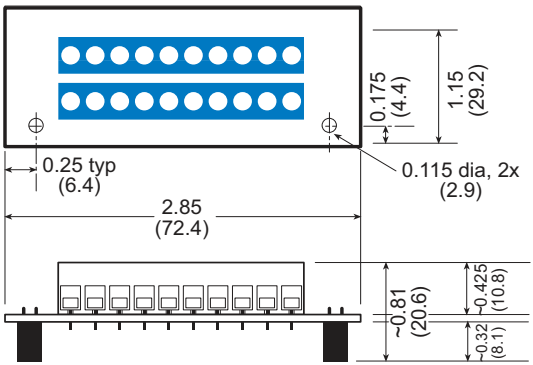


Figure C-9. FWT-A/D Dimensions

Figure C-10 shows the input channel assignments and pinouts for the FWT-A/D.

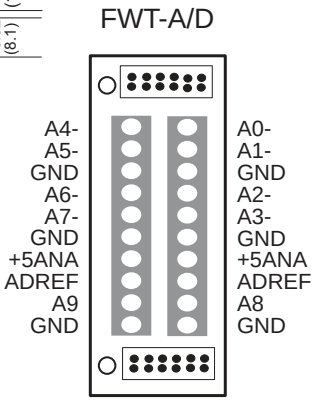


Figure C-10. FWT-A/D Pinouts

DIN Rails

The BL1700 and its expansion boards can be mounted using plastic standoffs to any flat surface that accepts screws. BL1700s can also be mounted in modular circuit-board holders and attached to DIN rail, a mounting system widely used for electrical components and controllers, as shown in Figure C-11.

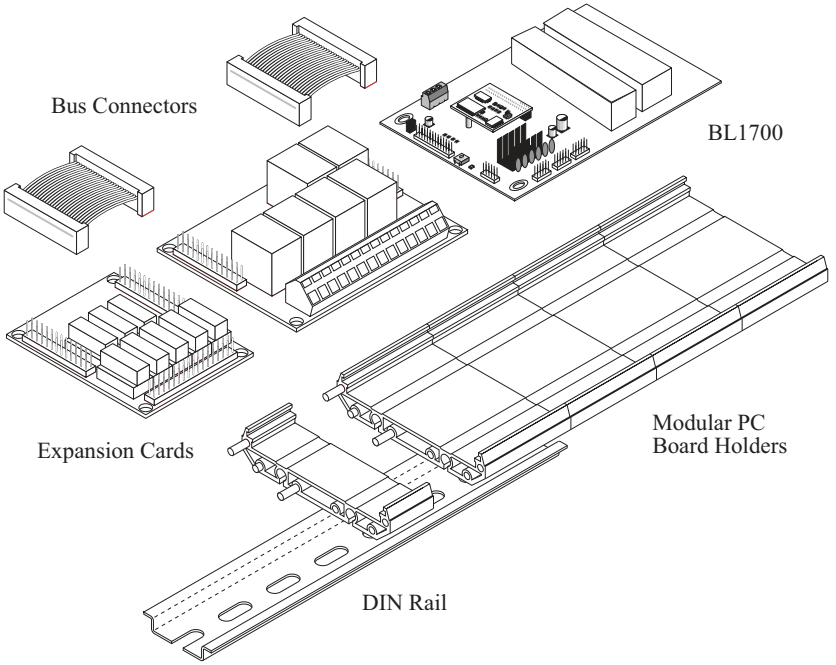


Figure C-11. Mounting BL1700 on DIN Rail

A DIN rail is a long metal rail. The BL1700 and other expansion boards slide snugly into modular, plastic printed-circuit board holders, which then snap onto the rail.

The BL1700 uses 110 mm circuit holders, which are available from Phoenix Contact. Z-World sells 75 mm circuit holders in multiples of lengths of 11.25 mm, 22.5 mm, or 45 mm for its expansion boards.

APPENDIX D: ***SINKING AND SOURCING DRIVERS***

BL1700 Series Sinking and Sourcing Outputs

The BL1700 Series controllers are normally supplied with ULN2803 sinking drivers. Figure D-1 shows a typical sinking driver output configuration.

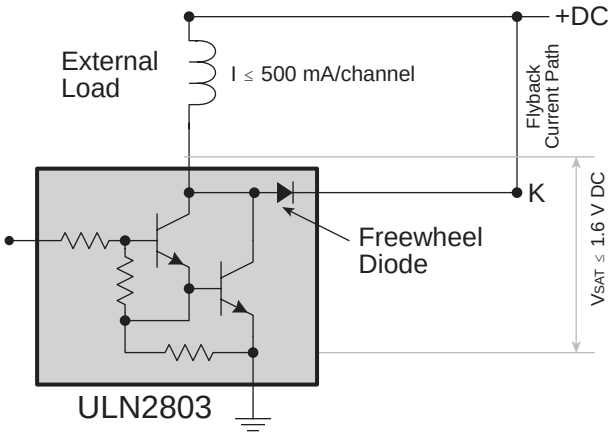


Figure D-1. Sinking Driver Output

Figure D-2 shows the jumper configurations for a sinking driver output.

SINKING DRIVER JUMPER SETTINGS

Figure D-2. Sinking Driver Jumper Configurations

Sourcing outputs are possible by replacing the factory-installed sinking driver chips with sourcing output drivers (UDN2985). The UDN2985 sourcing driver chip is capable of sourcing a maximum of 75 mA per output.

Figure D-3 shows a typical sourcing driver output.

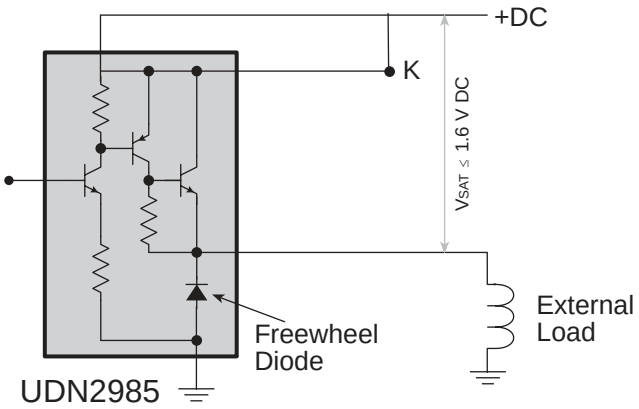


Figure D-3. Sourcing Driver Output

Figure D-4 shows the jumper configurations for a sourcing driver output.

SOURCING DRIVER JUMPER SETTINGS

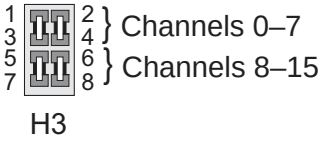


Figure D-4. Sourcing Driver Jumper Configurations

Z-World also offers all BL1700 Series controllers for quantity orders with factory-installed sourcing drivers. For ordering information, call your Z-World Sales Representative at (530) 757-3737.

Installing Sourcing Drivers

Figure D-5 shows the location of the drivers and headers with jumpers to be changed.

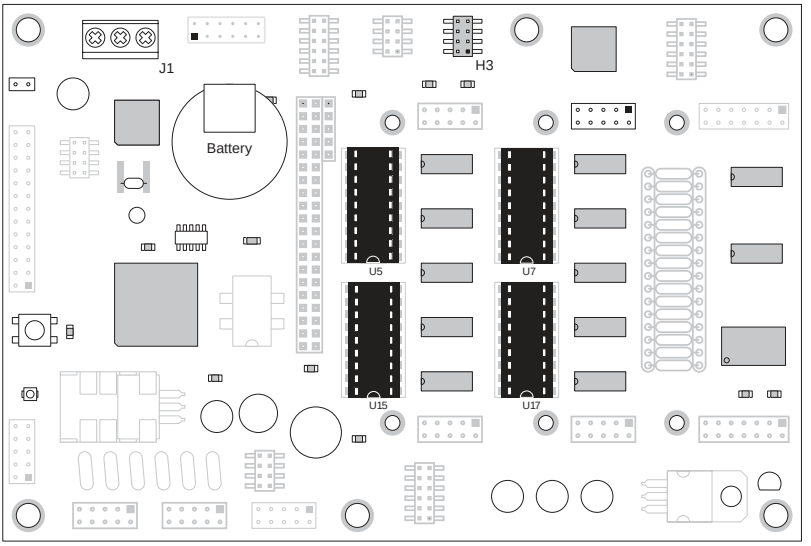


Figure D-5. U5, U7, U15 and U17 Locations of Sinking Drivers

Pay particular attention to the orientation of the jumpers when changing the driver output from sinking to sourcing. Exercise caution when installing sourcing drivers in the field.

1. Be sure power is removed from the controller.
2. Remove the ULN2803 sinking drivers from the IC sockets. Note that regular BL1700s have two ULN2803 chips (at U7 and U17) and only BL1700s that have been customized for more than 16 outputs will have chips at U5 and U15.
3. Install the jumpers on header H3 for the sourcing configuration, as shown in Figure D-4. Note the location of pin number 1 in Figure D-5.
4. Install UDN2985 sourcing driver chips into the IC sockets.

> Be sure the jumper settings conform to what is specified. Failure to install jumpers correctly may damage your controller.

TTL/CMOS Outputs

Z-World also offers TTL- or CMOS-compatible outputs for the BL1700 Series controllers. Input and output channels may be configured independently in any combination. However, the functionality of each input is not independent; the inputs are still characterized in groups of four or eight.

Z-World offers all BL1700 Series controllers in quantity with factory-installed TTL- or CMOS-compatible outputs. For ordering information, call your Z-World Sales Representative at (530) 757-3737.

Using Output Drivers

The common supply for all eight channels supplied by a driver chip is called "K," and is labeled as such on the BL1700's terminals. "K" must be powered up to allow proper operation.

The "K" connection performs two vital functions to the high-voltage driver circuitry on the BL1700.

1. "K" supplies power to driver circuitry inside the driver chip.
2. "K" also allows a diode internal to the driver chip to "snub" voltage transients produced during the inductive kick associated with switching inductive loads. (Relays, solenoids, and speakers are examples of inductive loads.)

Long leads may present enough induction to also produce large potentially damaging voltage transients. The anodes of the protection diodes for each channel are common, and so only one voltage supply can be used for all high-voltage driver loads.

The following points summarize the functions of "K."

- K provides power to the driver chip circuitry.
- K provides "clamping" for all high-voltage driver loads.
- It is mandatory to connect K regardless of whether sourcing or sinking.
- The load's supply must have a common ground with all other supplies in your system.
- All loads must use same supply voltage.

Refer to Figure D-6 and Figure D-7 when connecting K.

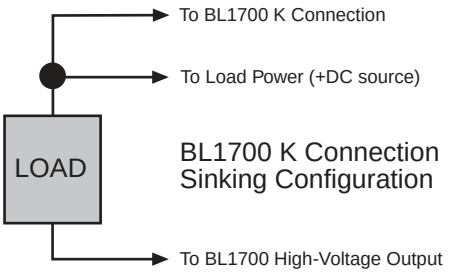


Figure D-6. BL1700 K Connections (Sinking Configuration)

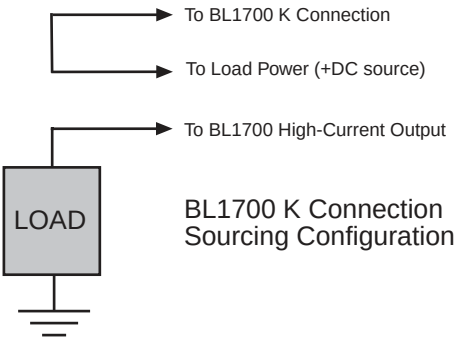


Figure D-7. BL1700 K Connections (Sourcing Configuration)

K must be connected to the power supply used for the high-voltage load. See Figure D-6 and Figure D-7.

APPENDIX E: PLCBUS

Appendix E provides the pin assignments for the PLCBus, describes the registers, and lists the software drivers.

PLCBus Overview

The PLCBus is a general-purpose expansion bus for Z-World controllers. The PLCBus is available on the BL1200, BL1600, BL1700, PK2100, PK2200, and PK2600 controllers. The BL1000, BL1100, BL1300, BL1400, and BL1500 controllers support the XP8300, XP8400, XP8600, and XP8900 expansion boards using the controller's parallel input/output port. The BL1400 and BL1500 also support the XP8200 and XP8500 expansion boards. The ZB4100's PLCBus supports most expansion boards, except for the XP8700 and the XP8800. The SE1100 adds expansion capability to boards with or without a PLCBus interface.

Table E-1 lists Z-World's expansion devices that are supported on the PLCBus.

Table E-1. Z-World PLCBus Expansion Devices

Device	Description
EXP-A/D12	Eight channels of 12-bit A/D converters
SE1100	Four SPDT relays for use with all Z-World controllers
XP8100 Series	32 digital inputs/outputs
XP8200	"Universal Input/Output Board" —16 universal inputs, 6 high-current digital outputs
XP8300	Two high-power SPDT and four high-power SPST relays
XP8400	Eight low-power SPST DIP relays
XP8500	11 channels of 12-bit A/D converters
XP8600	Two channels of 12-bit D/A converters
XP8700	One full-duplex asynchronous RS-232 port
XP8800	One-axis stepper motor control
XP8900	Eight channels of 12-bit D/A converters

Multiple expansion boards may be linked together and connected to a Z-World controller to form an extended system.

Figure E-1 shows the pin layout for the PLCBus connector.

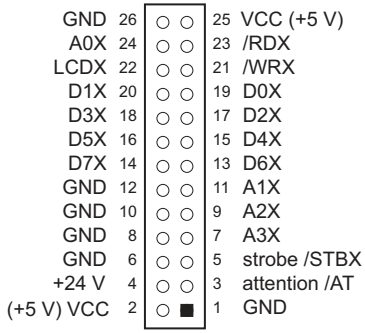


Figure E-1. PLCBus Pin Diagram

Two independent buses, the LCD bus and the PLCBus, exist on the single connector.

The LCD bus consists of the following lines.

- LCDX—positive-going strobe.
- /RDX—negative-going strobe for read.
- /WRX—negative-going strobe for write.
- A0X—address line for LCD register selection.
- D0X-D7X—bidirectional data lines (shared with expansion bus).

The LCD bus is used to connect Z-World's OP6000 series interfaces or to drive certain small liquid crystal displays directly. Figure E-2 illustrates the connection of an OP6000 interface to a controller PLCBus.

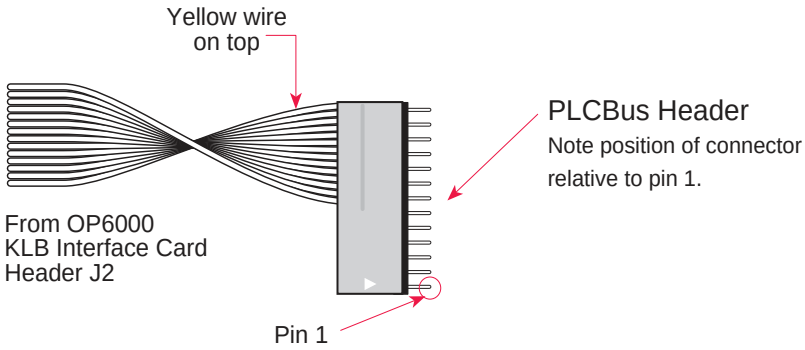


Figure E-2. OP6000 Connection to PLCBus Port

The PLCBus consists of the following lines.

- /STBX—negative-going strobe.
- A1X–A3X—three control lines for selecting bus operation.
- D0X–D3X—four bidirectional data lines used for 4-bit operations.
- D4X–D7X—four additional data lines for 8-bit operations.
- /AT—attention line (open drain) that may be pulled low by any device, causing an interrupt.

The PLCBus may be used as a 4-bit bus (D0X–D3X) or as an 8-bit bus (D0X–D7X). Whether it is used as a 4-bit bus or an 8-bit bus depends on the encoding of the address placed on the bus. Some PLCBus expansion cards require 4-bit addressing and others (such as the XP8700) require 8-bit addressing. These devices may be mixed on a single bus.

There are eight registers corresponding to the modes determined by bus lines A1X, A2X, and A3X. The registers are listed in Table E-2.

Table E-2. PLCBus Registers

Register	Address	A3	A2	A1	Meaning
BUSRD0	C0	0	0	0	Read data, one way
BUSRD1	C2	0	0	1	Read data, another way
BUSRD2	C4	0	1	0	Spare, or read data
BUSRESET	C6	0	1	1	Read this register to reset the PLCBus
BUSADR0	C8	1	0	0	First address nibble or byte
BUSADR1	CA	1	0	1	Second address nibble or byte
BUSADR2	CC	1	1	0	Third address nibble or byte
BUSWR	CE	1	1	1	Write data

Writing or reading one of these registers takes care of all the bus details. Functions are available in Z-World's software libraries to read from or write to expansion bus devices.

To communicate with a device on the expansion bus, first select a register associated with the device. Then read or write from/to the register. The register is selected by placing its address on the bus. Each device recognizes its own address and latches itself internally.

A typical device has three internal latches corresponding to the three address bytes. The first is latched when a matching BUSADR0 is detected. The second is latched when the first is latched and a matching BUSADR1 is detected. The third is latched if the first two are latched and a matching BUSADR2 is detected. If 4-bit addressing is used, then there are three 4-bit address nibbles, giving 12-bit addresses. In addition, a special register address is reserved for address expansion. This address, if ever used, would provide an additional four bits of addressing when using the 4-bit convention.

If eight data lines are used, then the addressing possibilities of the bus become much greater—more than 256 million addresses according to the conventions established for the bus.

Place an address on the bus by writing (bytes) to BUSADR0, BUSADR1 and BUSADR2 in succession. Since 4-bit and 8-bit addressing modes must coexist, the lower four bits of the first address byte (written to BUSADR0) identify addressing categories, and distinguish 4-bit and 8-bit modes from each other.

There are 16 address categories, as listed in Table E-3. An "x" indicates that the address bit may be a "1" or a "0."

Table E-3. First-Level PLCBus Address Coding

First Byte	Mode	Addresses	Full Address Encoding
– – – – 0 0 0 0 – – – – 0 0 0 1 – – – – 0 0 1 0 – – – – 0 0 1 1	4 bits × 3	256 256 256 256	0000 xxxx xxxx 0001 xxxx xxxx 0010 xxxx xxxx 0011 xxxx xxxx
– – – x 0 1 0 0 – – – x 0 1 0 1 – – – x 0 1 1 0 – – – x 0 1 1 1	5 bits × 3	2,048 2,048 2,048 2,048	x0100 xxxxx xxxxx x0101 xxxxx xxxxx x0110 xxxxx xxxxx x0111 xxxxx xxxxx
– – x x 1 0 0 0 – – x x 1 0 0 1	6 bits × 3	16,384 16,384	xx1000 xxxxxx xxxxxx xx1001 xxxxxx xxxxxx
– – x x 1 0 1 0	6 bits × 1	4	xx1010
– – – – 1 0 1 1	4 bits × 1	1	1011 (expansion register)
x x x x 1 1 0 0	8 bits × 2	4,096	xxxx1100 xxxxxxxx
x x x x 1 1 0 1	8 bits × 3	1M	xxxx1101 xxxxxxxx xxxxxxxx
x x x x 1 1 1 0	8 bits × 1	16	xxxx1110
x x x x 1 1 1 1	8 bits × 1	16	xxxx1111

This scheme uses less than the full addressing space. The mode notation indicates how many bus address cycles must take place and how many bits are placed on the bus during each cycle. For example, the 5 × 3 mode means three bus cycles with five address bits each time to yield 15-bit addresses, not 24-bit addresses, since the bus uses only the lower five bits of the three address bytes.

Z-World provides software drivers that access the PLCBus. To allow access to bus devices in a multiprocessing environment, the expansion register and the address registers are shadowed with memory locations known as *shadow registers*. The 4-byte shadow registers, which are saved at predefined memory addresses, are as follows.

SHBUS0	SHBUS0+1	SHBUS1 SHBUS0+2	SHBUS1+1 SHBUS0+3
Bus expansion	BUSADR0	BUSADR1	BUSADR2

Before the new addresses or expansion register values are output to the bus, their values are stored in the shadow registers. All interrupts that use the bus save the four shadow registers on the stack. Then, when exiting the interrupt routine, they restore the shadow registers and output the three address registers and the expansion registers to the bus. This allows an interrupt routine to access the bus without disturbing the activity of a background routine that also accesses the bus.

To work reliably, bus devices must be designed according to the following rules.

1. The device must not rely on critical timing such as a minimum delay between two successive register accesses.
2. The device must be capable of being selected and deselected without adversely affecting the internal operation of the controller.

Allocation of Devices on the Bus

4-Bit Devices

Table E-4 provides the address allocations for the registers of 4-bit devices.

Table E-4. Allocation of Registers

A1	A2	A3	Meaning
000j	000j	xxxj	digital output registers, 64 registers 64 × 8 = 512 1-bit registers
000j	001j	xxxj	analog output modules, 64 registers
000j	01xj	xxxj	digital input registers, 128 registers 128 × 4 = 512 input bits
000j	10xj	xxxj	analog input modules, 128 registers
000j	11xj	xxxj	128 spare registers (customer)
001j	xxxj	xxxj	512 spare registers (Z-World)

j controlled by board jumper
x controlled by PAL

Digital output devices, such as relay drivers, should be addressed with three 4-bit addresses followed by a 4-bit data write to the control register. The control registers are configured as follows

bit 3	bit 2	bit 1	bit 0
A2	A1	A0	D

The three address lines determine which output bit is to be written. The output is set as either 1 or 0, according to D. If the device exists on the bus, reading the register drives bit 0 low. Otherwise bit 0 is a 1.

For digital input, each register (BUSRD0) returns four bits. The read register, BUSRD1, drives bit 0 low if the device exists on the bus.

8-Bit Devices

Z-World's XP8700 and XP8800 expansion boards use 8-bit addressing. Refer to the ***XP8700 and XP8800*** manual.

Expansion Bus Software

The expansion bus provides a convenient way to interface Z-World's controllers with expansion boards or other specially designed boards. The expansion bus may be accessed by using input functions. Follow the suggested protocol. The software drivers are easier to use, but are less efficient in some cases. Table E-5 lists the libraries.

Table E-5. Dynamic C PLCBus Libraries

Library Needed	Controller
DRIVERS.LIB	All controllers
EZIOTGPL.LIB	BL1000
EZIOLGPL.LIB	BL1100
EZIOMGPL.LIB	BL1400, BL1500
EZIOPLC.LIB	BL1200, BL1600, PK2100, PK2200, ZB4100
EZIOPLC2.LIB	BL1700
PBUS_TG.LIB	BL1000
PBUS_LG.LIB	BL1100, BL1300
PLC_EXP.LIB	BL1200, BL1600, PK2100, PK2200

There are 4-bit and 8-bit drivers. The 4-bit drivers employ the following calls.

- **`void eioResetPlcBus()`**

  Resets all expansion boards on the PLCBus. When using this call, make sure there is sufficient delay between this call and the first access to an expansion board.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`void eioPlcAdr12( unsigned addr )`**

  Specifies the address to be written to the PLCBus using cycles BUSADR0, BUSADR1, and BUSADR2.

  PARAMETER: **`addr`** is broken into three nibbles, and one nibble is written in each BUSADR*x* cycle.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`void set16adr( int adr )`**

  Sets the current address for the PLCBus. All read and write operations access this address until a new address is set.

  PARAMETER: **`adr`** is a 16-bit physical address. The high-order nibble contains the value for the expansion register, and the remaining three 4-bit nibbles form a 12-bit address (the first and last nibbles must be swapped).

  LIBRARY: **`DRIVERS.LIB`**.

- **`void set12adr( int adr )`**

  Sets the current address for the PLCBus. All read and write operations access this address until a new address is set.

  PARAMETER: **`adr`** is a 12-bit physical address (three 4-bit nibbles) with the first and third nibbles swapped.

  LIBRARY: **`DRIVERS.LIB`**.

- **`void eioPlcAdr4( unsigned addr )`**

  Specifies the address to be written to the PLCBus using only cycle BUSADR2.

  PARAMETER: **`addr`** is the nibble corresponding to BUSADR2.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`void set4adr( int adr )`**

  Sets the current address for the PLCBus. All read and write operations access this address until a new address is set.

  A 12-bit address may be passed to this function, but only the last four bits will be set. Call this function only if the first eight bits of the address are the same as the address in the previous call to **`set12adr`**.

  PARAMETER: **`adr`** contains the last four bits (bits 8–11) of the physical address.

  LIBRARY: **`DRIVERS.LIB`**.

- **`char _eioReadD0( )`**

  Reads the data on the PLCBus in the BUSADR0 cycle.

  RETURN VALUE: the byte read on the PLCBus in the BUSADR0 cycle.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`char _eioReadD1( )`**

  Reads the data on the PLCBus in the BUSADR1 cycle.

  RETURN VALUE: the byte read on the PLCBus in the BUSADR1 cycle.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`char _eioReadD2( )`**

  Reads the data on the PLCBus in the BUSADR2 cycle.

  RETURN VALUE: the byte read on the PLCBus in the BUSADR2 cycle.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`char read12data( int adr )`**

  Sets the current PLCBus address using the 12-bit **`adr`**, then reads four bits of data from the PLCBus with BUSADR0 cycle.

  RETURN VALUE: PLCBus data in the lower four bits; the upper bits are undefined.

  LIBRARY: **`DRIVERS.LIB`**.

- **`char read4data( int adr )`**

  Sets the last four bits of the current PLCBus address using adr bits 8–11, then reads four bits of data from the bus with BUSADR0 cycle.

  PARAMETER: adr bits 8–11 specifies the address to read.

  RETURN VALUE: PLCBus data in the lower four bits; the upper bits are undefined.

  LIBRARY: **`DRIVERS.LIB`**.

- **`void _eioWriteWR( char ch)`**

  Writes information to the PLCBus during the BUSWR cycle.

  PARAMETER: ch is the character to be written to the PLCBus.

  LIBRARY: **`EZIOPLC.LIB`**, **`EZIOPLC2.LIB`**, **`EZIOMGPL.LIB`**.

- **`void write12data( int adr, char dat )`**

  Sets the current PLCBus address, then writes four bits of data to the PLCBus.

  PARAMETER: **`adr`** is the 12-bit address to which the PLCBus is set.

  `dat` (bits 0–3) specifies the data to write to the PLCBus.

  LIBRARY: **`DRIVERS.LIB`**.

- **`void write4data( int address, char data )`**

  Sets the last four bits of the current PLCBus address, then writes four bits of data to the PLCBus.

  PARAMETER: **`adr`** contains the last four bits of the physical address (bits 8–11).

  `dat` (bits 0–3) specifies the data to write to the PLCBus.

  LIBRARY: **`DRIVERS.LIB`**.

The 8-bit drivers employ the following calls.

- **`void set24adr( long address )`**

  Sets a 24-bit address (three 8-bit nibbles) on the PLCBus. All read and write operations will access this address until a new address is set.

  PARAMETER: **`address`** is a 24-bit physical address (for 8-bit bus) with the first and third bytes swapped (low byte most significant).

  LIBRARY: **`DRIVERS.LIB`**.

- **`void set8adr( long address )`**

  Sets the current address on the PLCBus. All read and write operations will access this address until a new address is set.

  PARAMETER: **`address`** contains the last eight bits of the physical address in bits 16–23. A 24-bit address may be passed to this function, but only the last eight bits will be set. Call this function only if the first 16 bits of the address are the same as the address in the previous call to **`set24adr`**.

  LIBRARY: **`DRIVERS.LIB`**.

- **`int read24data0( long address )`**

  Sets the current PLCBus address using the 24-bit address, then reads eight bits of data from the PLCBus with a BUSRD0 cycle.

  RETURN VALUE: PLCBus data in lower eight bits (upper bits 0).

  LIBRARY: **`DRIVERS.LIB`**.

- **`int read8data0( long address )`**

  Sets the last eight bits of the current PLCBus address using address bits 16–23, then reads eight bits of data from the PLCBus with a BUSRD0 cycle.

  PARAMETER: **`address`** bits 16–23 are read.

  RETURN VALUE: PLCBus data in lower eight bits (upper bits 0).

  LIBRARY: **`DRIVERS.LIB`**.

- **`void write24data( long address, char data )`**

  Sets the current PLCBus address using the 24-bit address, then writes eight bits of data to the PLCBus.

  PARAMETERS: **`address`** is 24-bit address to write to.

  `data` is data to write to the PLCBus.

  LIBRARY: **`DRIVERS.LIB`**.

- **`void write8data( long address, char data )`**

  Sets the last eight bits of the current PLCBus address using address bits 16–23, then writes eight bits of data to the PLCBus.

  PARAMETERS: **`address`** bits 16–23 are the address of the PLCBus to write.

  `data` is data to write to the PLCBus.

  LIBRARY: **`DRIVERS.LIB`**.

APPENDIX F:
SERIAL INTERFACE BOARD 2

Appendix F provides technical details and baud rate configuration data for Z-World's Serial Interface Board 2 (SIB2).

Introduction

The SIB2 is an interface adapter used to program the BL1700. The SIB2 is contained in an ABS plastic enclosure, making it rugged and reliable. The SIB2 enables the BL1700 to communicate with Dynamic C via the Z180's clocked serial I/O (CSI/O) port, freeing the BL1700's serial ports for use by the application during programming and debugging.

The SIB2's 8-pin cable plugs into the target BL1700's processor through an aperture in the backplate, and a 6-conductor RJ-12 phone cable connects the SIB2 to the host PC. The SIB2 automatically selects its baud rate to match the communication rates established by the host PC (9600, 19,200, 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 BL1700 (which also resets the SIB2), then select **Reset Target** from Dynamic C.

Chapter 2 provides detailed information on connecting the SIB2 to the BL1700.

The SIB2 receives power and resets from the target BL1700 via the 8-pin connector J1. Therefore, do not unplug the SIB2 from the target BL1700 while power is applied. To do so could damage both the BL1700 and the SIB2; additionally, the target may reset.

The SIB2 consumes approximately 60 mA from the +5 V supply. The target-system current consumption therefore increases by this amount while the SIB2 is connected to the BL1700.

When the BL1700 is powered up or reset with the SIB2 attached, it is automatically in the program mode. To operate the BL1700 in the run mode, remove power, disconnect the SIB2, and re-apply power to the BL1700.

Never connect or disconnect the SIB2 with power applied to the BL1700.

External Dimensions

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

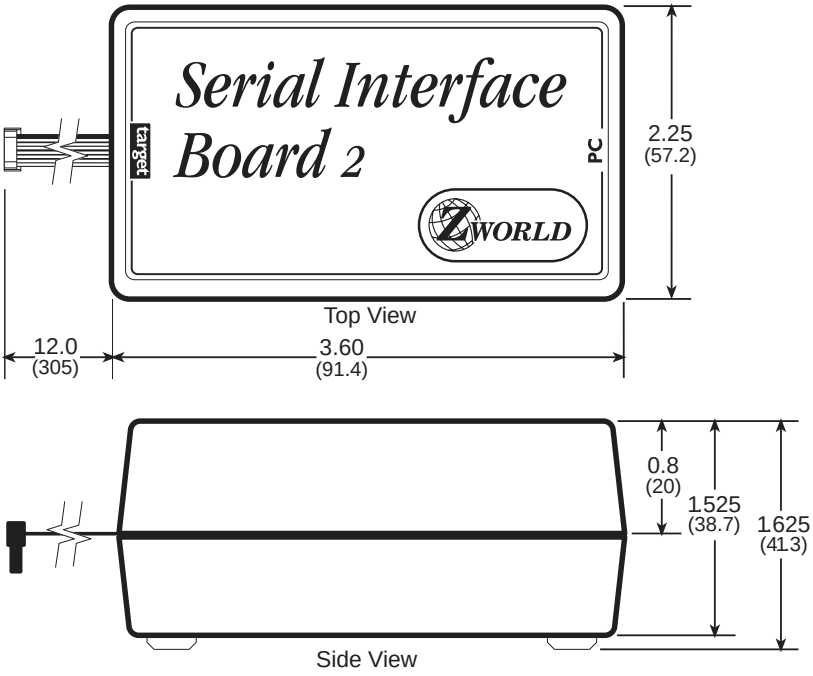


Figure F-1. SIB2 External Dimensions

APPENDIX G: ADVANCED TOPICS

Appendix G provides more advanced information to help the user needing to implement special applications. The following topics are included.

- Power Management
- Memory Map
- Interrupts
- Flash EPROM
- Pulse-Width Modulation Software

Power Management

Power Failure Detection Circuitry

Figure G-1 shows the power fail detection circuitry of the BL1700.

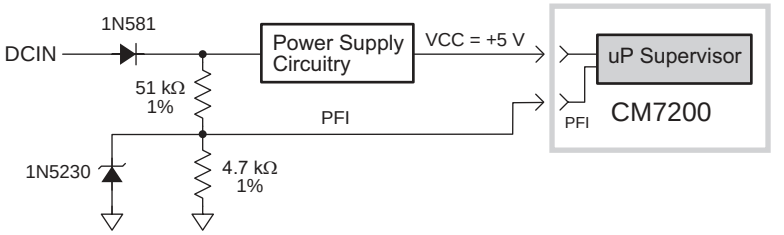


Figure G-1. BL1700 Power-Failure Detection Circuit

Power Failure Sequence of Events

Figure G-2 shows the events that occur as the input power fails.

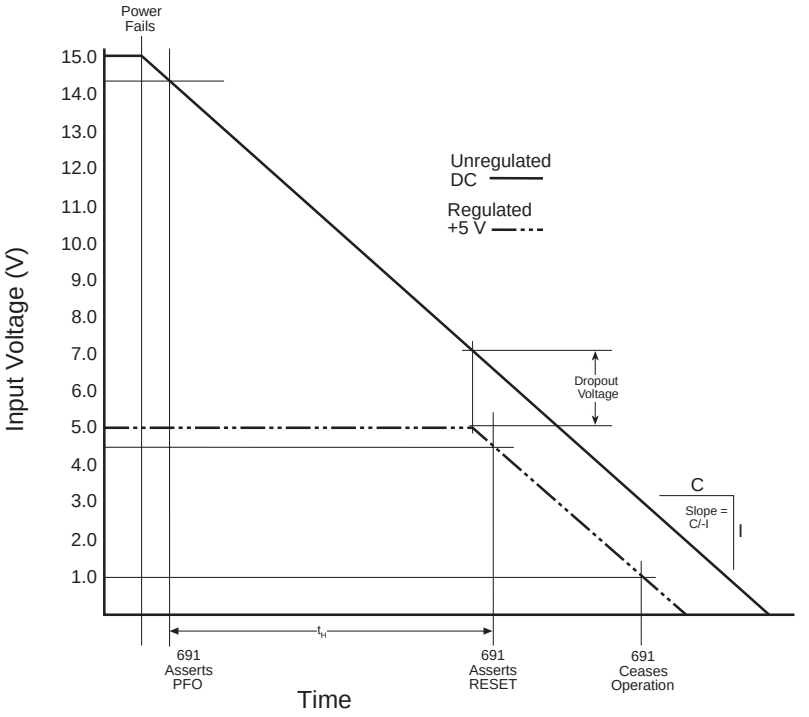


Figure G-2. Power Failure Sequence

1. The power-management IC triggers a power-fail /NMI (nonmaskable interrupt) when the DC input voltage falls within the range of 14.44 V to 14.72 V DC.
2. At some point, the raw input voltage will not be sufficient for the regulator to provide 5 V DC to the BL1700 due to dropout voltage. At that point the regulated output begins to drop. The power-management IC triggers a reset when the regulated 5 V DC output falls within the range of 4.50 V to 4.75 V DC. This causes the power-fail routine to be invoked. The power-fail routine can be used to store important state data.

Tip Use a power supply with a large capacitance if you need to increase the holdup time. This will provide additional time for the BL1700 to execute a safe shutdown.

3. The power management IC switches power for the time/date clock and SRAM to the lithium backup battery when the regulated voltage falls below the battery voltage of approximately 3 V DC.
4. The power management IC keeps the system in reset until the regulated voltage drops below 1 V DC. At this point the power-management IC ceases operating. By this time, the portion of the circuitry not battery-backed has already ceased functioning.

The ratio of your power supply's output capacitor's value to your circuit's current draw determines the actual holdup time.

In order to improve the performance of the power-failure NMI circuit, we have added some hysteresis to the power-failure comparator on the core module by adding a resistor, R21, between the comparator input and output pins. R21 can be found on the 175-1093 versions of the core module. The hysteresis prevents the comparator from switching rapidly—and therefore generating multiple interrupts—when the input voltage is falling slowly. Once the comparator switches (DC IN falls to approximately 3.9 V), this feedback holds the input (PFI) low and prevents further interrupts from being generated. At this point, the 3.3 V regulator still has sufficient voltage to keep the processor operating, so that an interrupt service routine can perform shutdown tasks and "tidying up" before the Vcc line fails. The comparator will not turn the output (PFO) high until DC IN has risen to about 5 V. The hysteresis will also help prevent any system oscillation in adverse power supplies/loading situations.

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 BL1700 may dip below 14.44 V DC. The interrupt routine performs a shutdown. This shutdown turns off the relay, clearing the problem. However, if the cause of the

overload persists, the system oscillates, alternately experiencing an overload and then resetting. Using a power supply with a sufficiently large current capacity will correct this problem.

If you remove the power cable abruptly from the BL1700 side, then only the capacitors on the board provide power, reducing computing time to a few microseconds. These times can vary considerably depending on system configuration and loads on the BL1700 power supplies.

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

Memory Map

Input/Output Select Map

The Dynamic C library functions **IBIT**, **ISET** and **IRES** in the **BIOS.LIB** library allow bits in the I/O registers to be tested, set, and cleared. The I/O addresses need to be 16-bit addresses.

Z180 Internal Input/Output Register Addresses 0x00-0x3F

The internal registers for the I/O devices built into to the Z180 processor occupy the first 40 (hex) addresses of the I/O space. These addresses are listed in Table G-1.

Table G-1. Z180 Internal I/O Registers Addresses 0x00–0x3F

Address	Name	Description
0x00	CNTLA0	Serial Channel 0, Control Register A
0x01	CNTLA1	Serial Channel 1, Control Register A
0x02	CNTLB0	Serial Channel 0, Control Register B
0x03	CNTLB1	Serial Channel 1, Control Register B
0x04	STAT0	Serial Channel 0, Status Register
0x05	STAT1	Serial Channel 1, Status Register
0x06	TDR0	Serial Channel 0, Transmit Data Register
0x07	TDR1	Serial Channel 1, Transmit Data Register
0x08	RDR0	Serial Channel 0, Receive Data Register
0x09	RDR1	Serial Channel 1, Receive Data Register
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
0x11–0x13	—	Reserved
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

continued…

Table G-1. Z180 Internal I/O Registers Addresses 0x00–0x3F (concluded)

Address	Name	Description
0x18	FRC	Free-running counter
0x19–0x1F	—	Reserved
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
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	—	Reserved
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	—	Reserved
0x36	RCR	Refresh Control Register
0x37	—	Reserved
0x38	CBR	MMU Common Base Register
0x39	BBR	MMU Bank Base Register
0x3A	CBAR	MMU Common/ Bank Area Register
0x3B–0x3D	—	Reserved
0x3E	OMCR	Operation Mode Control Register
0x3F	ICR	I/O Control Register

BL1700 Peripheral Addresses

Table G-2 lists the addresses that control I/O devices external to the Z180 processor.

Table G-2. BL1700 External I/O Device Registers

Address	Name	R/W	Function
0x4000	EN485A	W	D0 = RS-485 Channel 1 Enable
0x4040	ENDI1	R	D0-D7 = Digital Input[00–15]
0x4040	TE485B	W	D0 = RS-485 Channel B Transmit Enable
0x4042	ENDI2	R	D0-D7 = Digital Input[16–31]
0x4042	TE485A	W	D0 = RS-485 Channel A Transmit Enable
0x40C0	ADEOC	R	D0 = ADC end of conversion
0x40D0	ADOUT	R	D0 = ADC output
0x40D0	ADIN	W	D2 = ADC instruction
0x40D0	ADCS	W	D1 = ADC chip select
0x40D0	ADCLK	W	D0 = ADC clock
0x40E0	LCD	R/W	PLCBus LCD line
0x40F0	/STB	R/W	PLCBus strobe
0x4100	/ENHV1	W	Digital Output [00–07]
0x4108	/ENHV2	W	Digital Output [08–15]
0x4110	/ENHV3	W	Digital Output [16–23]
0x4118	/ENHV4	W	Digital Output [24–31]
0x4120	SCC	R/W	Serial Channel B, Control
0x4121		R/W	Serial Channel A, Control
0x4122		R/W	Serial Channel B, Data
0x4123		R/W	Serial Channel A, Data
0x4142	LED	W	D0 = LED status
0x417F		R	D7 = Run/Program Mode, D6–D0 = ID

Refer to the Zilog or Hitachi ***User's Manual*** for more information about the Z80180/Z180 or the 64180 MPU internal I/O register map.

Epson 72423 Timer Registers 0x4180–0x418F

Table G-3 lists the Epson 72423 timer registers.

Table G-3. Epson 72423 Timer Registers 0x4180–0x418F

Address	Name	Data Bits	Description
4180	SEC1	D0–D7	seconds, units
4181	SEC10	D0–D7	seconds, tens
4182	MIN1	D0–D7	minutes, units
4183	MIN10	D0–D7	minutes, tens
4184	HOUR1	D0–D7	hours, units
4185	HOUR10	D0–D7	hours, tens
4186	DAY1	D0–D7	days, units
4187	DAY10	D0–D7	days, tens
4188	MONTH1	D0–D7	months, units
4189	MONTH10	D0–D7	months, tens
418A	YEAR1	D0–D7	years, units
418B	YEAR10	D0–D7	years, tens
4180C	WEEK	D0–D7	weeks
418D	TREGD	D0–D7	Register D
418E	TREGE	D0–D7	Register E
418F	TREGF	D0–D7	Register F

Interrupts

The BL1700 provides user access to two level-sensitive interrupts. The interrupts are shared with onboard peripherals such as the PLCBus port and serial channels. If these peripherals are not used, then external devices may use these interrupts. Header H1 provides connections to the /INT0 and /INT1 interrupt lines of the Z180 processor according to the pinout shown in Figure G-3. Header J4 provides jumper connections to allow external connection to the /INT0 and /INT1 signals.

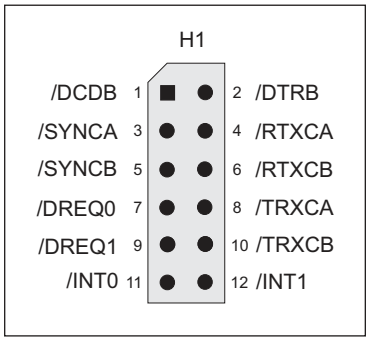


Figure G-3. Pinouts for Optional Header H1

See Chapter 3, "BL1700 Hardware," for information on jumper settings for header J4.

Z-World can install a header at header location H1 in production quantities. For more information, call your Z-World Sales Representative at (530) 757-3737.

Interrupts can be enabled or disabled by including the following commands in your code.

- Interrupt 0 "ON" (enabled)

```
~    outport(ITC,inport(ITC) | 1)
```

- Interrupt 1 "ON" (enabled)

```
~    outport(ITC,inport(ITC) | 2)
```

- Interrupt 0 "OFF" (disabled)

```
~    outport(ITC,inport(ITC) & 0xfe)
```

- Interrupt 1 "OFF" (disabled)

```
~    outport(ITC,inport(ITC) & 0xfd)
```

Interrupt Service Routines

Interrupt service routines are packets of code that the processor jumps to and executes when it receives an interrupt request.

Refer to the Dynamic C manuals for instructions on writing interrupt service routines.

Refer to the Zilog ***Z80180/Z180 User's Manual*** (available from Z-World) for complete details on using Z180 interrupts.

Interrupt Vectors

To "vector" an interrupt to a user function in Dynamic C, a directive such as the following is used.

```
#INT_VEC 0x10 myfunction
```

This example causes the interrupt at offset 10H (Serial Port 1 of the Z180) to invoke the function **myfunction()**. The function must be declared with the **interrupt** keyword.

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

Table G-4 provides the interrupt vectors for various Z180 internal devices.

Table G-4. Interrupt Vectors for Z180 Internal Devices

Address	Name	Description
0x00	**INT1_VEC**	/INT1
0x02	**INT2_VEC**	/INT2
0x04	**PRT0_VEC**	PRT timer Channel 0
0x06	**PRT1_VEC**	PRT timer Channel 1
0x08	**DMA0_VEC**	DMA Channel 0
0x0A	**DMA1_VEC**	DMA Channel 1
0x0C	**CSIO_VEC**	Clocked Serial I/O
0x0E	**SER0_VEC**	Asynchronous Serial Port Channel 0
0x10	**SER1_VEC**	Asynchronous Serial Port Channel 1

Table G-5 lists the interrupt priorities.

Table G-5. Interrupt Priorities

Interrupt Priorities	
(Highest Priority)	Trap (Illegal Instruction)
	NMI (Nonmaskable Interrupt)
	INT 0 (Maskable Interrupt, Level 0)
	INT 1 (Maskable Interrupt, Level 1)
	INT 2 (Maskable Interrupt, Level 2)
	PRT Timer Channel 0
	PRT Timer Channel 1
	DMA Channel 0
	DMA Channel 1
	Clocked Serial I/O
	Serial Port 0
(Lowest Priority)	Serial Port 1

Jump Vectors

Jump vectors are similar to interrupt vectors, except that instead of loading the address of the interrupt routine from the interrupt vector, these interrupts cause a jump directly to the address of the vector, which contains a jump instruction to the interrupt routine. This is an example of a jump vector.

```
0x66 nonmaskable power-failure interrupt
```

Because nonmaskable interrupts can be used for Dynamic C communication, the interrupt vector for power failure is normally stored just in front of the Dynamic C program. Store a vector there by using this compiler directive.

```
#JUMP_VEC NMI_VEC name
```

The Dynamic C communication routines jump to this vector when a power failure causes the NMI rather than a serial interrupt.

Flash EPROM

Simulated EEPROM

The BL1700 uses a section of the flash EPROM to simulate EEPROM. The size of the simulated EEPROM is 512 bytes (not Kbytes). Locations 0x02 through 0x3D are used to store the analog input calibration constants. The rest of the simulated EEPROM is free for use by the application. These functions are used to read/write from/to the simulated EEPROM.

- **`int ee_rd( int address )`**

  Reads and returns data from flash EPROM storage location **`address`**. The function returns –1 if it is unable to read data.

  LIBRARY: **`BIOS.LIB`**

- **`int ee_wr( int address, int data )`**

  Writes data to flash EPROM storage location **`address`**. The function returns –1 if it is unablae to write data.

  LIBRARY: **`BIOS.LIB`**

Other Flash EPROM Software

- **`int WriteFlash( unsigned long physical_addr, char *buf, int count )`**

  Writes **`count`** number of bytes pointed to by **`buf`** to the flash EPROM absolute data location **`physical_adr`**. Allocate the data location by declaring the byte arrays as initialized arrays or declare an initialized **`xdata`** array. If byte array is declared, conert logical memory to physical memory with **`phy_adr(array)`**. For initialized **`xdata`**, you can pass the array name directly.

  PARAMETERS: **`physical_adr`** is the absolute data location in the flash EPROM.

  `*buf` is a pointer to the bytes to write.

  `count` is the number of bytes to write.

  RETURN VALUES:

  0 if **`WriteFlash`** is okay.

  –1 if the flash EPROM is not in use.

  –2 if **`physical_addr`** is inside the BIOS area.

  –3 if **`physical_addr`** is within the symbol area or the simulated EEPROM area.

  –4 if **`WriteFlash`** times out.

  LIBRARY: **`DRIVERS.LIB`**

Flash EPROM is rated for 10,000 writes. In practice, flash EPROM has performed for up to 100,000 writes. Z-World recommends that any writes to the flash EPROM be made by the programmer rather than automatically by the software to maximize the life of the flash EPROM.

Pulse-Width Modulation (PWM) Software

PWM Addressing Detail

The driver of the PWM on the BL1700 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 edge detection mode. The simple interface previously described in Chapter 4 provides PWM support for output 0 to output 3. If the application requires more PWM channels or require 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 number of modulated channels, frequency, and resolution.

1. Determine the number of channels, frequency, and 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. The calculations in this section are made with the assumption that all channels have the same frequency and resolution.

The clock output from communication port 1 (CKA1) must have a frequency,

$$f_1 = N_{ch} \times f_w \times R_w \ ,$$

where which 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, **`_eioSetupAO1st`**, makes the following assumptions.

$$N_{ch} = 4$$

$$f_1 = 76{,}800 \text{ Hz}$$

$$R_w = 256$$

Consequently, $f_w = 76{,}800 \text{ Hz}/(4 \times 256) = 75 \text{ Hz}$.

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

Memory must be allocated to store the waveform pattern.

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 four bits select one of the high-current outputs, output 0 through output 6. The least 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 output 0 and output 1 are the only pulse-width modulated channels, a composite transition consists of two bytes. The first byte specifies a possible transition for channel output 0. The second byte specifies a possible transition for channel output 1.

Let us assume the first byte in the composite transition corresponds to output 0, and the second byte corresponds to output 1.

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 output 0 is to have a 0.375 duty cycle, output 1 is to be at 0.75 duty cycle, and the resolution is 8 divisions per cycle, a simple wave form would be as follows.

CT1: turn on output 0, turn on output 1.

CT2: do nothing.

CT3: do nothing.

CT4: turn off output 0.

CT5: do nothing.

CT6: do nothing.

CT7: turn off output 1.

CT8: do nothing.

Go back to CT1.

Outputting the byte 0x01 turns on output 0, 0x00 turns off output 0, 0x03 turns on output 1, and 0x02 turns off output 1. The byte 0x0E is an "no-op" and it does nothing. The composite transitions (with no-ops) can be translated into the following byte sequence to be sent to the I/O address 0x4100.

CT1: 0x01, 0x03

CT2: 0x0E, 0x0E

CT3: 0x0E, 0x0E

CT4: 0x00, 0x0E

CT5: 0x0E, 0x0E

CT6: 0x0E, 0x0E

CT7: 0x0E, 0x02

CT8: 0x0E, 0x0E

Go back to CT1

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

0x01, 0x03, 0x0E, 0x0E, 0x0E, 0x0E, 0x00, 0x0E,
0x0E, 0x0E, 0x0E, 0x0E, 0x0E, 0x02, 0x0E, 0x0E

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

4. Set up the clock.

The DMA device transfer from memory to I/O port address 0x4100 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 transfer memory to I/O. Each transfer corresponds to one edge in the previous section.

Refer to the Zilog user's manual for more information on how to set up the CKA1 frequency for the Z80180/Z180 or to the Hitachi user's manual for the 64180 MPU.

The driver does include a function, **`dmapwmInit`**, that sets up the frequency of CKA1. The function is described later in this appendix.

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

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 device 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 device. 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 while it is 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.

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

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, **`_eioSetupAO1st()`** sets up f_1 = 76,800Hz and l_{wpb} = 4096. As a result, the application engineer must ensure $f_r \geq 37.5$ Hz.

6. Changing 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 effective 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 the "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". For example, in our example in the "set up the

waveform" section, if we wish to change the duty cycle of output 0 to 0.25, we change the waveform from

0x01, 0x03, 0x0E, 0x0E, 0x0E, 0x0E, 0x00, 0x0E,
0x0E, 0x0E, 0x0E, 0x0E, 0x0E, 0x02, 0x0E, 0x0E

to

0x01, 0x03, 0x0E, 0x0E, 0x00, 0x0E, 0x0E, 0x0E,
0x0E, 0x0E, 0x0E, 0x0E, 0x0E, 0x02, 0x0E, 0x0E

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 manner.

However, although 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, besides the waveform pattern buffer, the application program must maintain a duty cycle variable for each channel.

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 involved 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.

PWM Software

The functions shown below are second level functions that allow more PWM outputs. They are also more complex and require a more in-depth understanding of PWM and DMA generation. These functions are located in **EZIODPWM.LIB**.

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

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

  In other words, **dmapwmSetBuf** does the following: starting at the address pointed to by **pBufStart**, for **bufLength256** many 256-byte pages, change every **step** bytes to **outChar**.

  PARAMETERS: **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.

- **`void dmapwmSwBuf ( unsigned newBuf256 )`**

  In order to facilitate all-or-none duty cycle transitions, you should use two buffers. While one buffer is being used by the DMA mechanism 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 **`cka1rate`** * 19.2 kHz to /DREQ0. DMA Channel 0 would then perform memory to I/O transfer for each clock pulse falling edge.

  PARAMETERS: **`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, of type **`(char *)`**), the following expression should be passed to this parameter

  ```
  (unsigned)((xmadr(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 19.2 kHz units. Allowed numbers are 2, 4, and 8.

Sample Program

BL17PWM1.C and **BL17PWM2.C** are sample programs which show how to use the pulse width modulation feature using the functions listed above. They can be found in the Dynamic C **SAMPLES\BL17XX** directory.

The PWM functions use the Z180's built-in DMA hardware. Using this DMA-driven PWM limits the communication speed of the Z180's Serial Port 1 to 4800 bps, and the Z180 runs effectively at least 8% slower. In addition you must ensure your application calls **_eioBrdAORf** at least every 25 ms to refresh the drivers' period.

If necessary, call Z-World Technical Support at (530)757-3737 for assistance.

APPENDIX H: BATTERY

Appendix H provides information about the onboard lithium backup battery.

Battery Life and Storage Conditions

The battery on the BL1700 controller will provide at least 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 controller is unpowered and the static RAM size. Most systems are operated on a continuous basis, with the battery supplying power to the real-time clock and the static RAM 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 190 mA·h. At 25°C, the real-time clock draws 3 µA when idle, and the 32K SRAM draws 2 µA. If the BL1700 were unpowered 100 percent of the time, the battery would last 38, 000 hours (4.3 years).

To maximize the battery life, the BL1700 should be stored at room temperature in the factory packaging until field installation. Take care that the BL1700 is not exposed to extreme temperature, humidity, and/or contaminants such as dust and chemicals.

To ensure maximum battery shelf life, follow proper storage procedures. Replacement batteries should be kept sealed in the factory packaging at room temperature until installation. Protection against environmental extremes will help maximize battery life.

Replacing Soldered 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 the battery 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 Panasonic BR23251HG or its equivalent.

Battery Cautions

- Caution **(English)**

  There is a danger of explosion if the 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 Behandein der Batterie. Nur durch gleichen Typ oder vom Hersteller empfohlenen Ersatztyp ersetzen. Entsorgung der gebrauchten Batterien gemäb 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.

Symbols

A

B

C

D

E

F

H

I

J

K

L

M

O

P

R

S

T

U

W

X

Z

SCHEMATICS

This page is blank intentionally.