OP6000

Operator Interface

User's Manual

Revision D

OP6000 User's Manual

Part Number 019-0020 • Revision D
Last revised on April 13, 1999 • Printed in U.S.A.

Copyright

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.

Notice to Users

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

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

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

Company Address

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

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

TABLE OF CONTENTS

ABOUT THIS MANUAL

This manual provides instructions for installing, testing, configuring, and interconnecting the Z-World OP6000 Operator Interface.

Instructions are included to get you started using Dynamic C software programming functions. In addition to C and Dynamic C references, programming resources are referenced when necessary.

Assumptions

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

- Ability to design and engineer a target system that uses an OP6000 Operator Interface.
- 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

- Knowledge of basic Intel assembly language and architecture for controllers with an Intel386 EX processor.

For documentation from Intel, refer to the following texts:

Intel386 EX Embedded Microprocessor User's Manual
Intel386 SX Microprocessor Programmer's Reference Manual

Acronyms

Table 1 is a list of 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		High Voltage
	Caution	Tip	Tip
	Factory Default		

Conventions

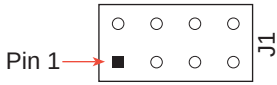
Table 3 lists and defines typographical conventions that may be used in this manual.

Table 3. Typographical 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.

Blank

CHAPTER 1: OVERVIEW

Chapter 1 introduces and describes components and supporting hardware for three OP6000 operator interfaces: OP6100, OP6200, and OP6300. The OP6400 operator interface is described in the ***OP6400 User's Manual***.

Introduction

An OP6000 series operator interface is ideal for applications that require inputting commands or data, reading messages, or monitoring system functions. Each member of the OP6000 series is outlined in the following list.

- The OP6100 and OP6110 have a 4 × 20 character LCD and a 4 × 6 keypad that does not require a plastic spacer. The OP6110 provides backlighting for the LCD.
- The OP6200 has a 4 × 40 character display and a 4 × 10 keypad.
- The OP6300 has a 256 × 64 graphic display and a 4 × 10 keypad.
- The OP6400 has a 256 × 128 graphic display, a 15 × 15 touch-screen overlay, and an RS-232 communication interface. The OP6400 operator interface is described in the ***OP6400 User's Manual***.

The OP6000 series operator interface connects to a Z-World controller through an LCD port. On most Z-World controllers, the LCD port is part of the PLCBus port. Figure 1-1 shows the layout of a typical interface card.

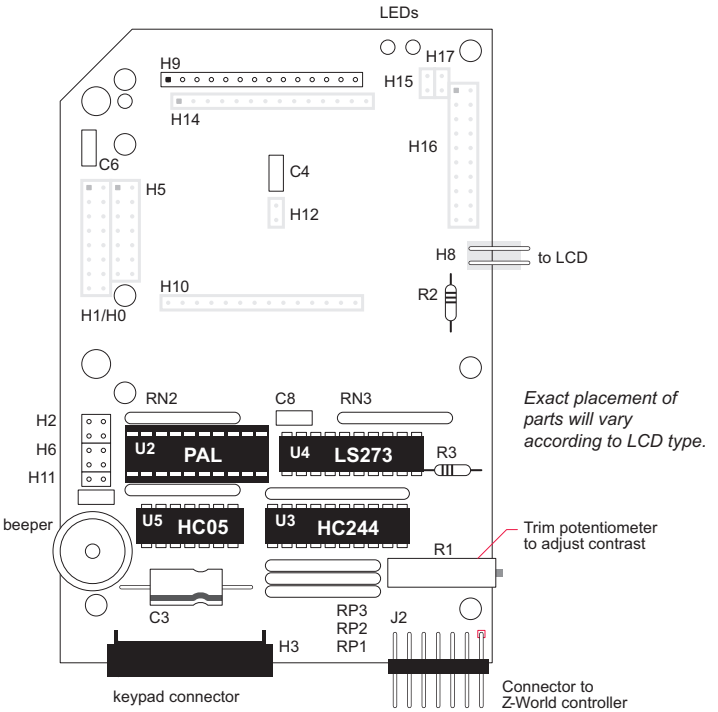


Figure 1-1. Typical OP6000 KLB Interface Card

OP6000 Components

The following main components make up a complete OP6000 package. Together, the components provide a complete LCD/keypad interface to the controller.

- *Liquid Crystal Display (LCD)*. Several sizes of character and graphics models are available. See the section on LCD support for specific types.
- *Keypad.* Two sizes of keypads are available, a 4 × 10 and a 4 × 6 version. All keys are completely user-definable.
- *Keypad Insert.* The keypad insert displays the symbols relating to each key's function. One standard insert is included with the keypad. Custom inserts are easily made.
- *Keypad Mask.* The glare-resistant top layer of an OP6000 overlay protects the LCD, keypad, and plate from dirt, fingerprints, and scratches. Two sizes are available to match the two available enclosures.
- *Interface Card (KLB).* The interface card connects the controller to the LCD and the keypad. Two types are available, one for character LCDs and one for graphic LCDs.
- *Enclosure.* The box that holds the controller and allows easy mounting of the LCD and keypad. With the enclosure, an entire OP6000 system can be mounted efficiently. A large (7.38″W × 8.38″H × 3.00″D) and small (5.25″W × 6.75″H × 2.31″D) version are available. The enclosures are made of molded glossy black phenolic MIL-M-14, CFG.
- *Aluminum Plate.* Covers the case and includes a window for viewing the display. Three types are available to match available enclosures to different LCD sizes. Chapter 2 includes mechanical drawings of each aluminum plate.
- *Plastic Spacer.* Sits between the aluminum plate and the keypad mask on the OP6200 and the OP6300. The OP6100 does not require a separate plastic spacer since the keypad provides one. The spacer fills the empty gap caused by the thickness of the keypad. A spacer is available to match the aluminum plate for the OP6200 and the OP6300. A mechanical drawing of each plastic spacer is shown in Chapter 2.

LCD Support

The KLB interface card was designed to connect to a wide range of commonly used displays. The Seiko line is the preferred choice for character LCDs since it is the closest to an industry standard.

The preferred graphic LCDs are from the Toshiba line. The OP6000 graphic interface is designed for displays with the onboard T6963C controller. When choosing a graphic display, be sure that it uses the T6963C controller.

Table 1-1 through Table 1-5 list the display modules known to connect directly to the KLB interface card. Other displays for the KLB interface card may be available. The header locations are shown in Figure 1-1.

Table 1-1. Supported Seiko Character LCDs

Model	Size	KLB Connection	H2 Jumper Configuration
L1671	1 × 16	H9	open
L1672	2 × 16	H5	1-3, 2-4
L1682	2 × 16	H9	open
L1692	2 × 16	H10	open
L1634	4 × 16	H9	open
L2032	2 × 20	H5	1-2, 3-4
L2034	4 × 20	H9	open
L2462	2 × 24	H5	1-2, 3-4
L4052	2 × 40	H5	1-2, 3-4
L4044	4 × 40	H1	1-3, 2-4

Table 1-2. Supported Samtron Character LCDs

Model	Size (dots)	KLB Connection	H2 Jumper Configuration
UC-404-01	40 × 4	H0	1-3, 2-4

Table 1-3. Supported Toshiba Graphic LCDs

Model	Size (dots)	KLB Connection	H2 Jumper Configuration
TLX-711	240 × 64	H16	open
TLX-1021	120 × 64	H16	open
TLX-1391	128 × 128	H16	open
TLX-1013	160 × 128	H16	open
TLX-1301	240 × 128	H16	open

Table 1-4. Supported Samtron Graphic LCDs

Model	Size (dots)	KLB Connection	H2 Jumper Configuration
UG-24B-03*	240 × 4	H16	open

* The UG-24B-03 is functionally identical to the Toshiba TLX-711.

Table 1-5. Supported Seiko Graphic LCD Controller Boards

Model	KLB Connection	H2 Jumper Configuration
SED1330*	H16	open

* Provides control over most clocked, 4-bit interface graphic modules.

Blank

CHAPTER 2: GETTING STARTED

Chapter 2 provides instructions for connecting a controller with an LCD interface card, a display, and a keypad. Separate instructions are provided for character displays and graphic displays.

Electrical Installation

The KLB interface card connects directly to character LCDs via a header and socket. A 20-conductor ribbon cable is provided to connect the KLB interface card to a graphic display. Two different multicolored ribbon cables are provided to connect either the character or the graphic KLB interface card with a controller. The cable with a 14-pin connector at each end is used with the BL1000 and the BL1000 controllers. The other cable with a 14-pin connector at one end and a 26-pin connector at the other is used for connections to the PLCBus on other controllers.

KLB Interface Card Connections

A header (H2) supports the different pinouts of the various LCDs. Since an incorrect configuration at H2 can damage an LCD, it is imperative that jumpers are correctly connected. Jumpers are set for the LCD supplied in OP6000 kits.

Refer to the LCD compatibility tables in Chapter 1, Table 1-1 through Table 1-5, for proper H2 settings when using another LCD.

Figure 2-1 shows the header locations on the KLB interface card.

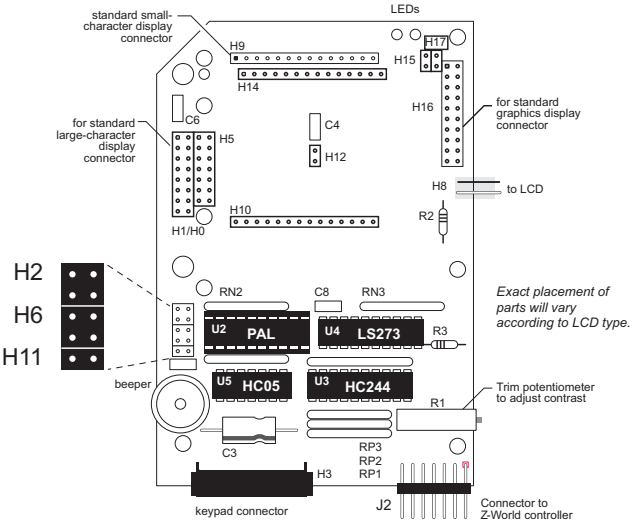


Figure 2-1. Header Locations

Always make sure power is off when connecting electronic devices.

The KLB interface card always connects to the controller from header J2, as shown in Figure 2-2.

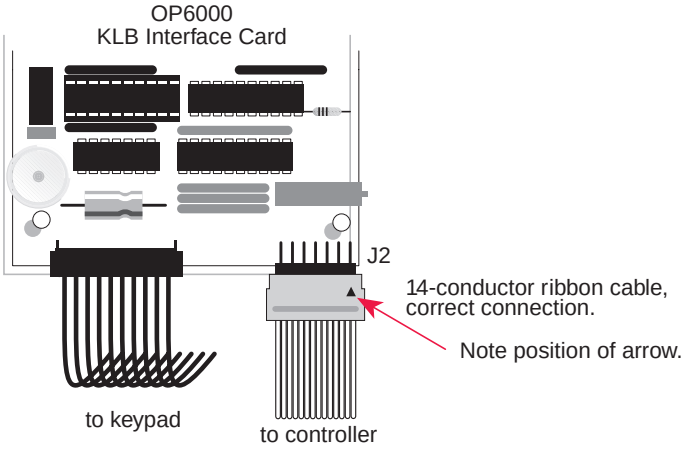


Figure 2-2. KLB Interface Card Connection

BL1000 Connections

Use the ribbon cable with two 14-pin connectors. Connect one end to header J2 on the KLB interface card (as shown in Figure 2-2). Connect the other end to the BL1000's LCD port (header J1) as shown in Figure 2-3.

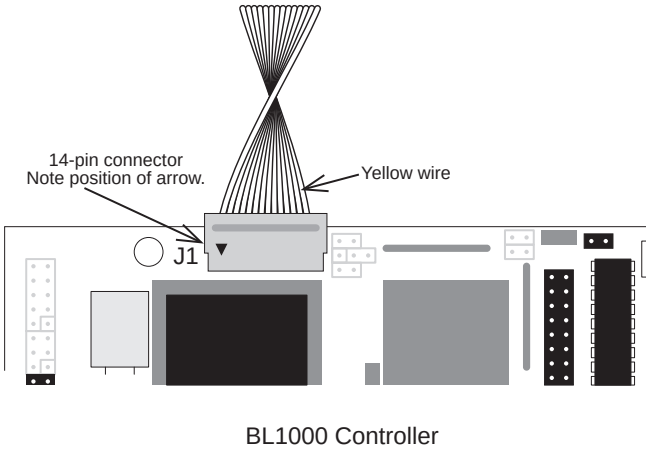


Figure 2-3. BL1000 Connections to OP6000 KLB Interface Card

BL1100 Connections

Use the ribbon cable with two 14-pin connectors. Connect one end to header J2 on the KLB interface card (as shown in Figure 2-2). Connect the other end to the BL1100's LCD port (header J6) as shown in Figure 2-4.

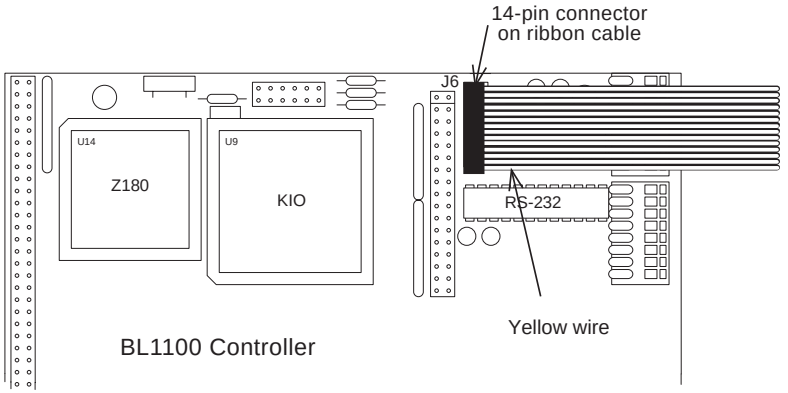


Figure 2-4. BL1100 Connections to OP6000 KLB Interface Card

PLCBus Connections

Use the ribbon cable with a 14-pin connector and a 26-pin connector to connect the OP6000 KLB interface card to any controller with a PLCBus. Connect the 14-pin connector to header J2 on the KLB interface card (as shown in Figure 2-2). Connect the 26-pin connector to the PLCBus header as shown in Figure 2-5.

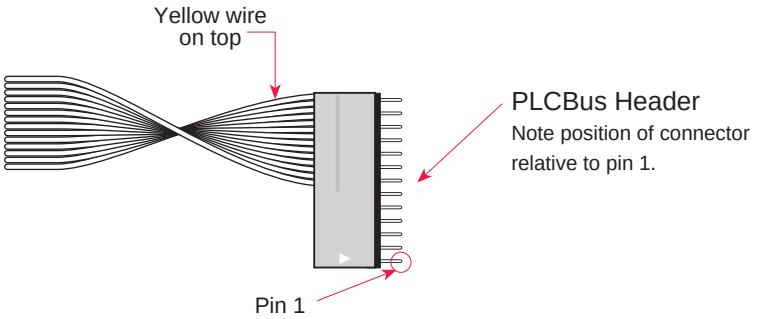


Figure 2-5. PLCBus Connections to OP6000 KLB Interface Card

An incorrect connection between the KLB interface card and the controller's PLCBus can damage both the interface card and the controller.

If expansion boards are attached to the controller, connect the OP6000 KLB interface card to the last expansion board.

PK2100 and PK2200 controllers have an LCD port header. ***Never*** connect the KLB interface card to this LCD port. If you do, you will damage the graphic display. ***Always*** connect the KLB interface card to the PLCBus when using a PK2100 or a PK2200 controller.

Character Display Installation

The KLB interface card is connected with the character LCD via a header and socket. Figure 2-6 illustrates how a 4 × 20 character display is connected to the KLB interface card. Other character displays connect with similar orientation, but use different connectors on the KLB interface card. The possible socket locations for character displays are H0, H1, H5, H9, and H10. The OP6000 is shipped with the correct sockets installed. Refer to the LCD support tables (Table 1-1 to Table 1-5, pages 1-4 and 1-5) to determine which location is used for other LCDs.

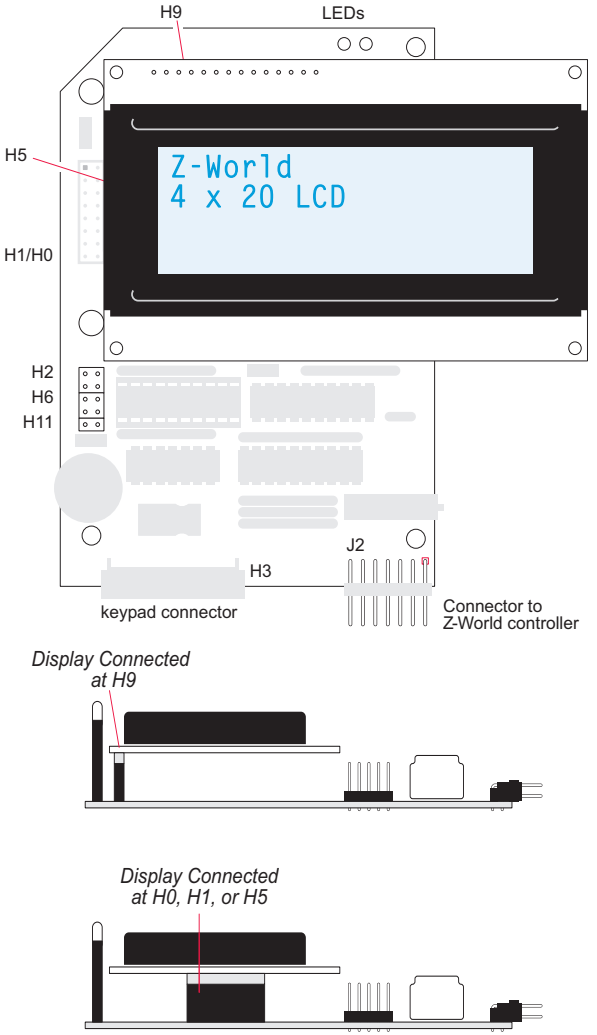


Figure 2-6. 4 × 20 Character Display Installation

Connect the KLB interface card to the controller from header J2, as shown in Figure 2-7. The KLB interface card is connection to a controller's PLCBus or LCD port as explained earlier in this chapter.

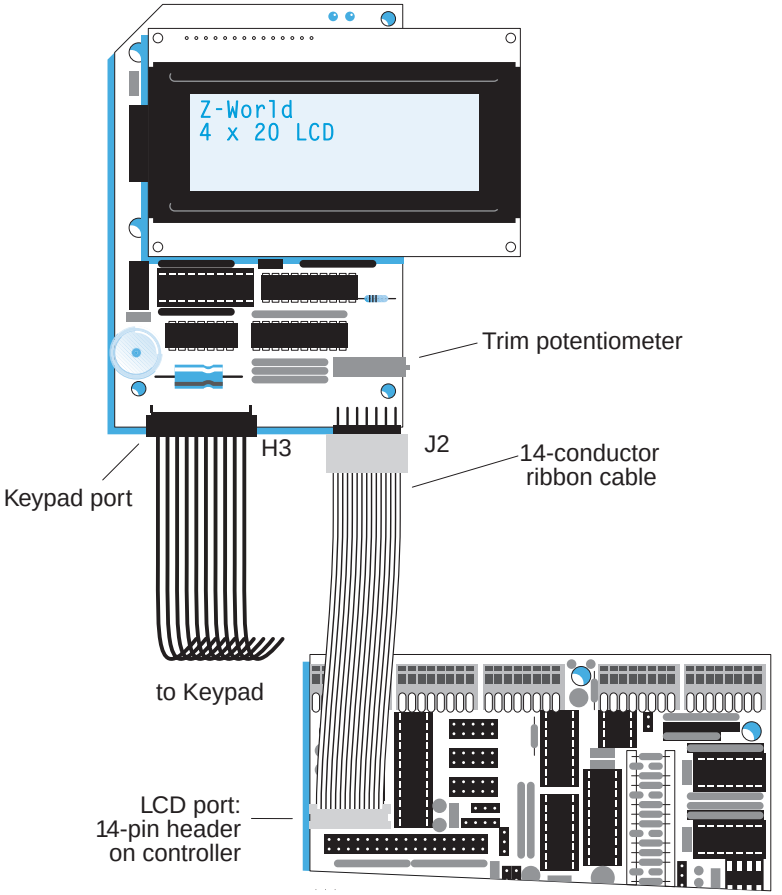


Figure 2-7. Standard 4 × 20 Character Display Connection

The keypad connects with an attached flat-flex cable into header H3 of the KLB interface card.

Once the electrical connections are complete, apply power to the controller and run a program that will display something on the LCD. If nothing appears on the LCD, try adjusting the trim potentiometer until the display is at the desired contrast.

Graphic Display Installation

The graphic display is connected through the 20-pin right-angle connector H16. A 20-conductor ribbon cable is provided with standard graphic OP6000 packages to connect the display to the KLB interface card via H16. Figure 2-8 shows an example of a standard graphic display package connected to a Z-World controller. The KLB interface card is connected to the controller PLCBus port or LCD port with the 14-conductor ribbon cable as it was for the character display.

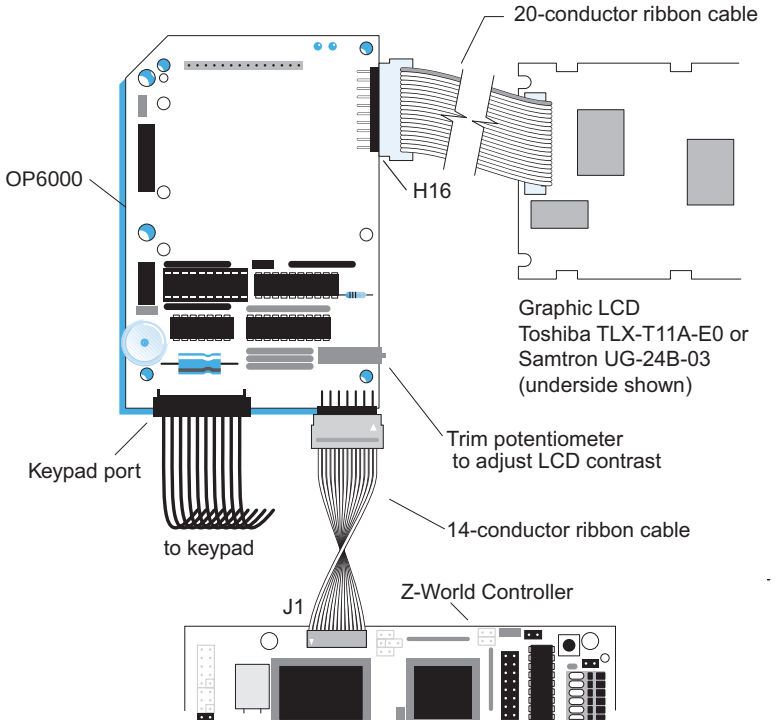


Figure 2-8. Standard Graphic Display Connection

Contrast Adjustment

The trim potentiometer on the KLB interface card controls the drive voltage to the LCD. This voltage is set prior to shipping and should work on initial operation. However, if the display is weak or is not visible, then adjust the trim potentiometer until the desired contrast is achieved. Run one of Z-World's packaged test programs to facilitate this adjustment prior to mounting the system.

LCD Backlighting

The KLB supports LED backlighting. LCDs using this type of backlighting may be purchased from Z-World.

For more information regarding backlighting, refer to ***Seiko Instruments LCD Character Modules Application Notes****, 2nd Ed.* pp. 22–23. Telephone: (213) 517-7770.

LED backlighting requires low voltage but high current (around 480 mA). This usually requires a current-limiting resistor and simple +5-V (separate) and ground connections. Z-World makes these connections for the customer when this option is requested. LED backlighting offers the advantage of a long life span (typically 50,000 h) and does not require the AC-DC inverter that EL lighting requires. The backlighting connections are shown in Figure 2-9.

The LED backlight usually consists of surface-mount LEDs attached to a shallow plastic tray.

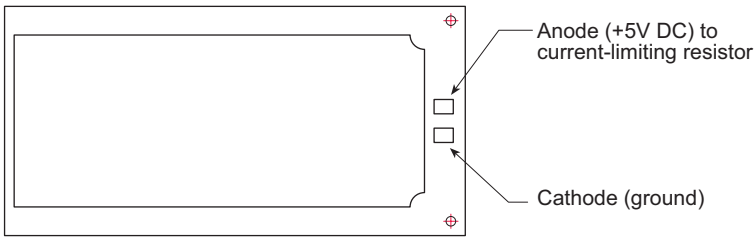


Figure 2-9. Typical Backlighting Connections

Enclosure Mounting

Two enclosures are available to house a complete controller-KLB interface card-LCD-keypad package. Refer to Figure 2-10 as a guide in mounting a system.

A controller can be mounted to the back of the enclosure by drilling appropriate holes in the casing. The aluminum plate has an aperture for feeding the keypad flex cable to the KLB interface card. The plate also has two holes at the bottom so that cables can be fed through.

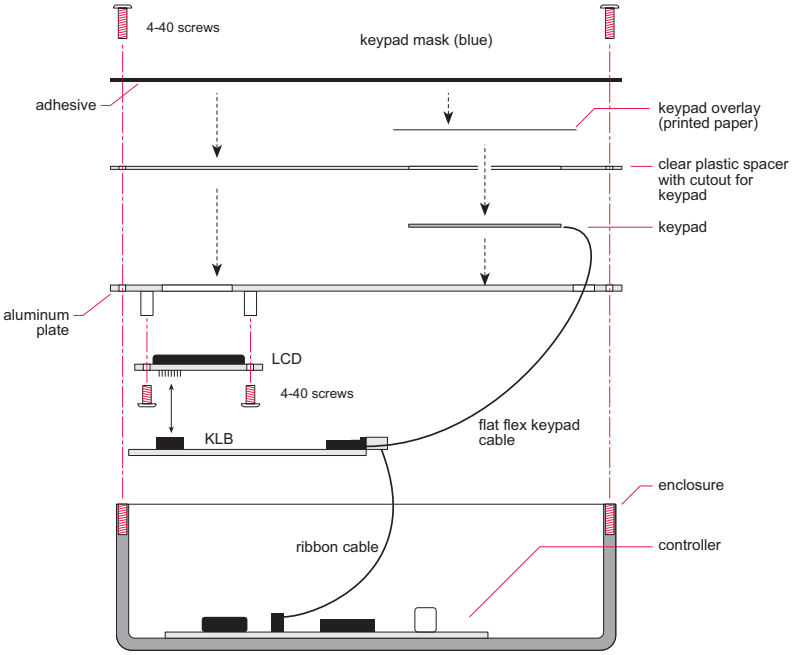


Figure 2-10. Enclosure Mounting

KLB Interface Card Plates and Spacers

Figure 2-11 through Figure 2-15 are mechanical drawings of the aluminum plates and plastic spacers available through Z-World. Each is shown as a set since they are intended to go together.

OP6100 Aluminum Plate

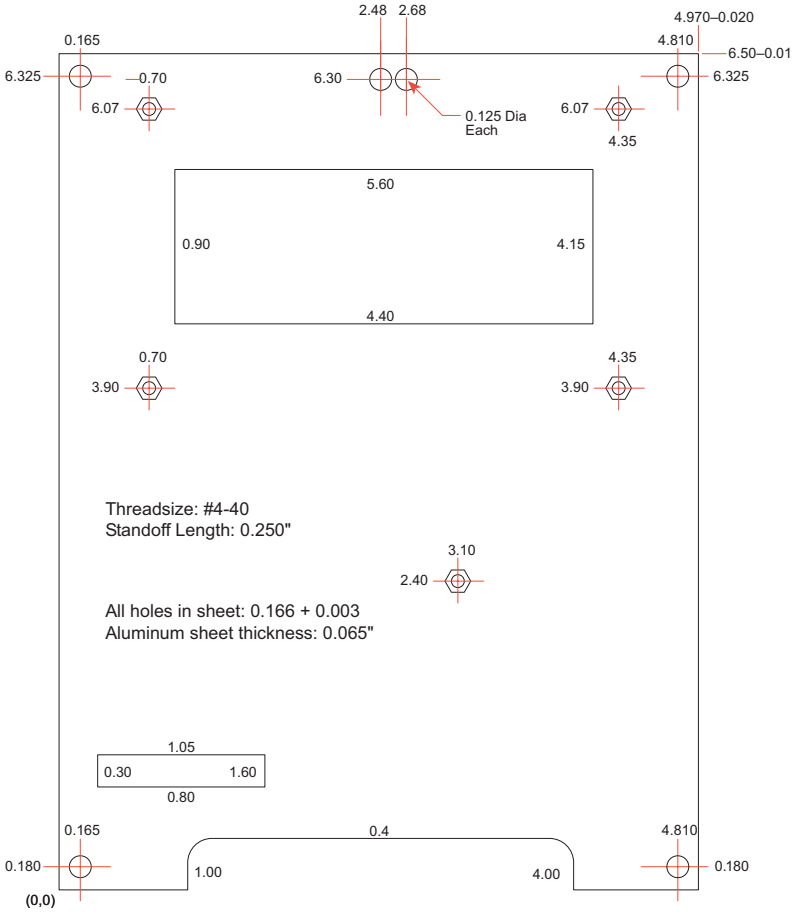


Figure 2-11. OP6100 Aluminum Plate

OP6200 Aluminum Plate and Plastic Spacer

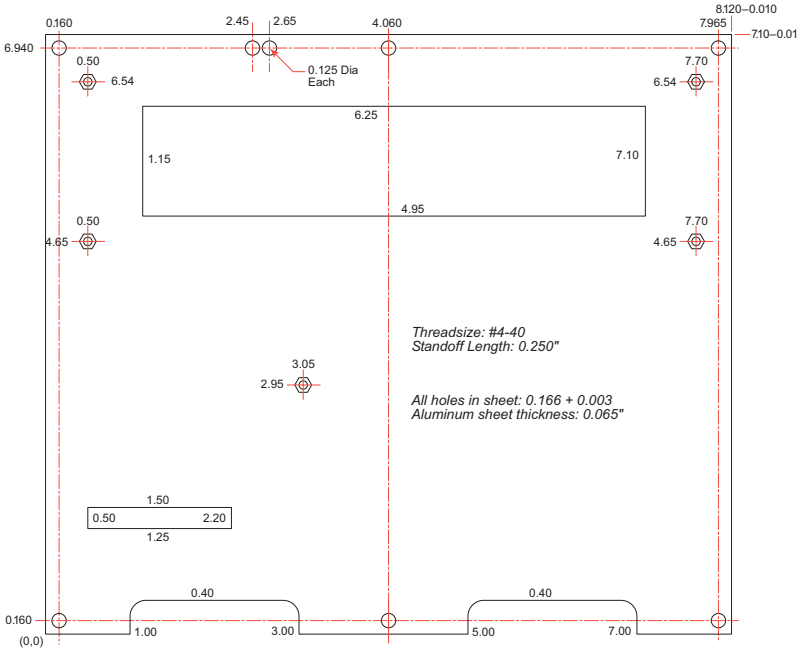


Figure 2-12. OP6200 Aluminum Plate

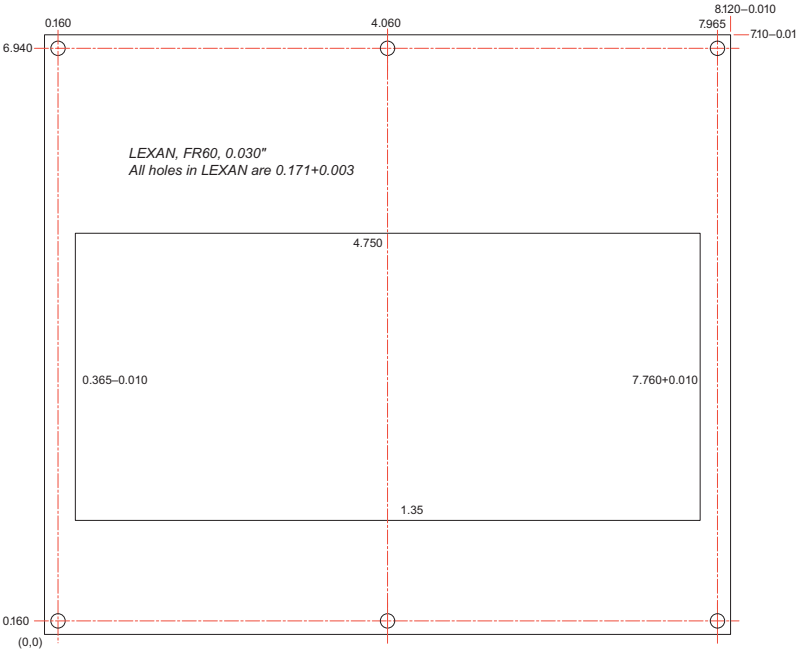


Figure 2-13. OP6200 Clear Plastic Spacer

OP6300 Aluminum Plate and Plastic Spacer

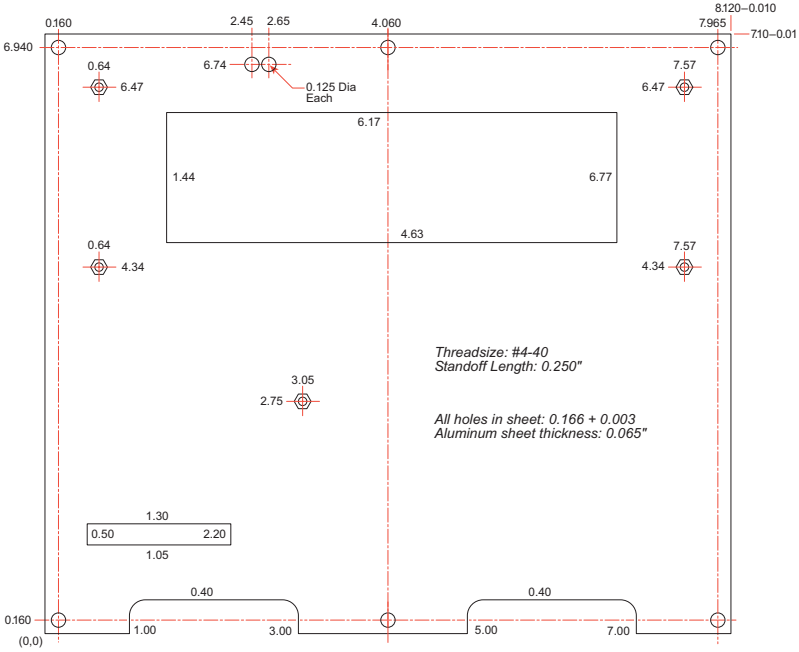


Figure 2-14. OP6300 Aluminum Plate

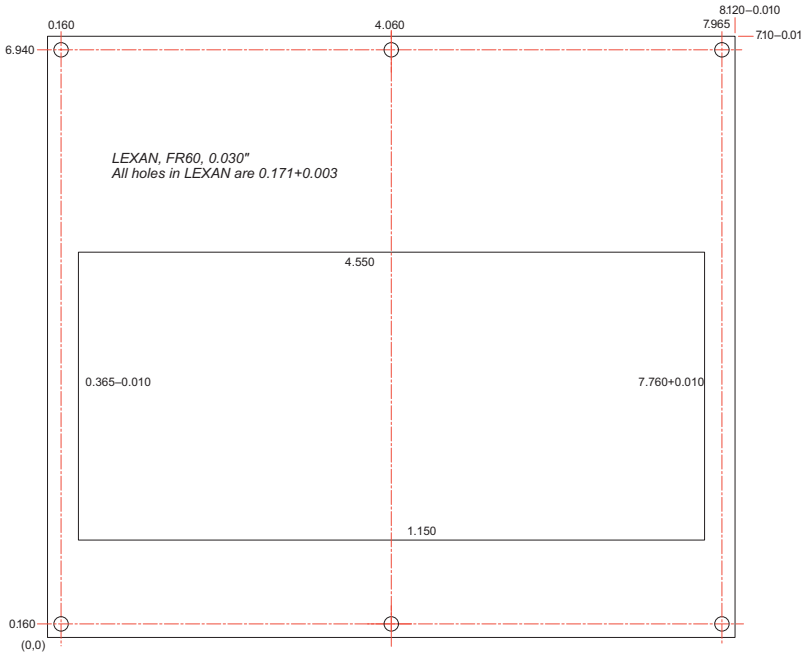


Figure 2-15. OP6300 Clear Plastic Spacer

Blank

CHAPTER 3: OPERATION

Chapter 3 provides an overview of how the active OP6000 components operate.

Display Operation

When the LCD has been addressed by the KLB interface card, commands may be sent to the LCD. These commands may be actual characters to display, or they may be escape sequences intended to execute an LCD function such as clearing the display or positioning the cursor.

The 40 × 4 character display acts as two separate 40 × 2 displays. Thus the software must distinguish which one to address when accessing the display.

Dynamic C functions, described in Chapter 4, are available to operate the LCD. Refer to the LCD data sheets when designing functions for a specific LCD. These data sheets are usually available directly from the vendors. The vendor telephone numbers and manual names for the LCDs available through Z-World are listed below.

Seiko Instruments (213) 517-7770.
LCD Character Modules Application Notes, 2nd Ed.

Toshiba Corporation (408) 739-0560.
ST LCD Modules

Keypad Operation

The keypad is a simple device that detects presses by making contact between a distinct pair of lines for each key (see Figure 3-1). The 4 × 10 keypad has 13 lines attached. The 4 × 6 keypad has 10 lines attached.

Column and row lines are addressed by one or more registers. Setting a column line to zero in an output register "enables" that column. A low-value reading for a row indicates that a keypress has been made *at that column* and *at that row.* Low value-readings for all columns and all rows gives a complete keypad scan. The KLB interface card employs this method to read the keypad.

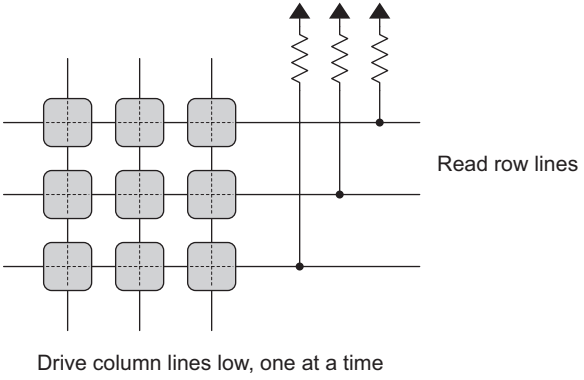


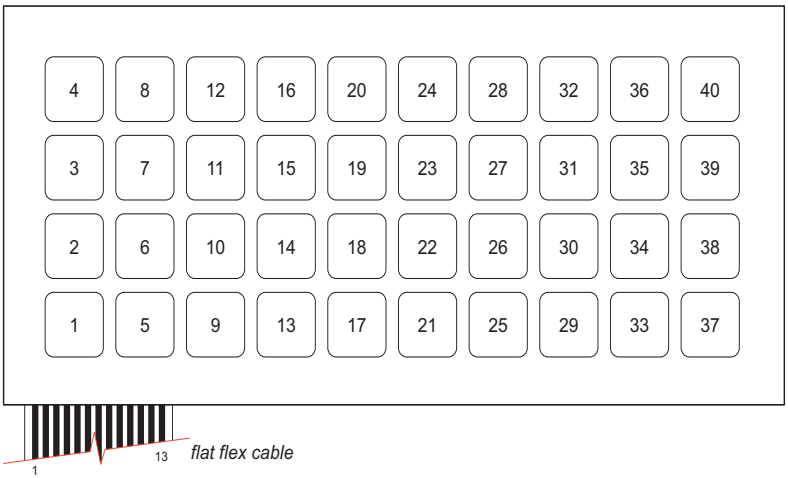
Figure 3-1. Keypad Read Operation

40-key Matrix-Scanning Keypad

Table 3-1 presents the termination schedules for the 40-key keypad. The numbers in the "Connection" column indicate the pair of wires in the ribbon cable that connect when the key number listed under "Key" is pressed. Figure 3-2 shows the keypad layout.

Table 3-1. 40-key Keypad Termination Schedule

Key	Connection	Key	Connection	Key	Connection
1	1-7	15	2-10	28	4-13
2	1-9	16	2-12	29	4-6
3	1-11	17	3-7	30	4-8
4	1-13	18	3-9	31	4-10
5	1-6	19	3-11	32	4-12
6	1-8	20	3-13	33	5-7
7	1-10	21	3-6	34	5-9
8	1-12	22	3-89	35	5-11
9	2-7	23	3-10	36	5-13
10	2-9	24	3-12	37	5-6
11	2-11	25	4-7	38	5-8
12	2-13	26	4-9	39	5-10
13	2-6	27	4-11	40	5-12
14	2-8				



Note: The keys are numbered for orientation; they do not come labeled

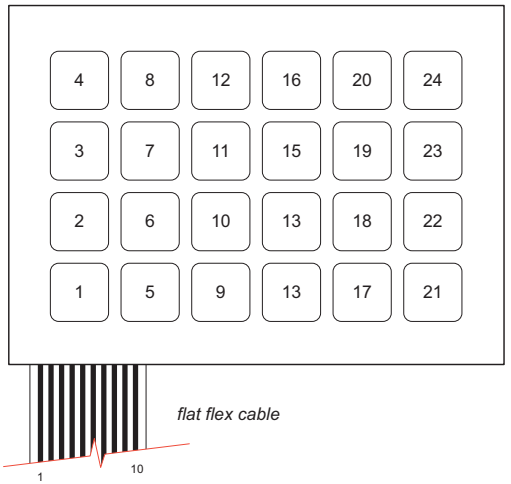
Figure 3-2. 40-key Keypad Layout

24-key Matrix-Scanning Keypad

Table 3-2 presents the termination schedules for the 24-key keypad. The numbers in the "Connection" column indicate the pair of wires in the ribbon cable that connect when the key number listed under "Key" is pressed. Figure 3-3 shows the keypad layout.

Table 3-2. 24-key Keypad Termination Schedule

Key	Connection	Key	Connection
1	1-5	13	1-8
2	2-5	14	2-8
3	3-5	15	3-8
4	4-5	16	4-8
5	1-6	17	1-9
6	2-6	18	2-9
7	3-6	19	3-9
8	4-6	20	4-9
9	1-7	21	1-10
10	2-7	22	2-10
11	3-7	23	3-10
12	4-7	24	4-10



Note: The keys are numbered for orientation; they do not come labeled

Figure 3-3. 24-key Keypad Layout

Beeper Operation

The beeper is controlled by a bit in a keypad register. The beeper sounds when this bit is zero. Dynamic C functions are available in **KDM.LIB** to control the beeper. The beeper functions are activated as described on page 4-9 of Chapter 4.

KLB Interface Card Operation

The KLB interface card takes commands sent through the LCD or PLCBus port of the controller, and routes them to the appropriate OP6000 device. Software first selects the appropriate device by writing a value to the KLB onboard PAL. The software then chooses the device and/or register the next command will access. Once the device is selected, the controller can access the device directly. Dynamic C functions, described in Chapter 4, control the operation of the KLB interface card.

Figure 3-4 illustrates the connections of the KLB interface card with the controller, the LCD, and the keypad.

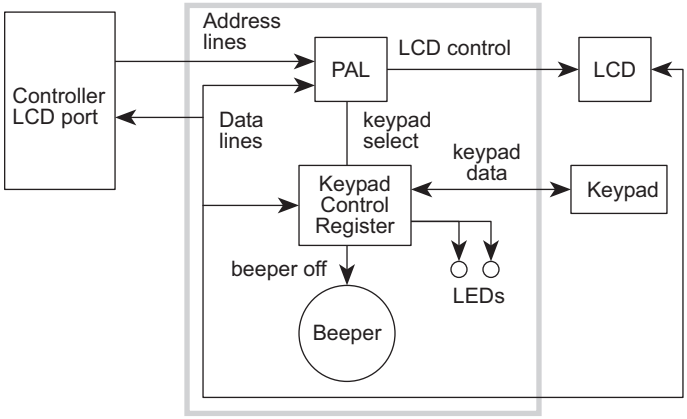


Figure 3-4. KLB Interface Connections

Blank

CHAPTER 4: SOFTWARE REFERENCE

The Dynamic C library **KDM.LIB** provides all software functions necessary to operate the OP6000 series interface without requiring detailed hardware knowledge.

Define Statements

The **KDM.LIB** functions rely on **#define** statements for correct operation. For programs used with OP6000 devices, always include the functions defined in Table 4-1 and be sure to include a **#use kdm.lib** statement to call **KDM.LIB**.

For example, when using a Samtron UC-404-01 character LCD with a 4 × 10 keypad, add the following lines to the program in order to use the KLB interface library functions:

```
#define KEYPAD_SIZE   40
#define LK_LINES       4
#define LK_COLS       40
```

Table 4-1. OP6000 Software Definitions

Name	Meaning	Standard Values and Application	
KEYPAD_SIZE	Keys on keypad	24	6 × 4 keypad
		40	10 × 4 keypad
LK_LINES	Rows of characters	4	40 × 4 LCD
		2	Any 2-line LCD
LK_COLS	Character columns	40	Any 40-column LCD
		20	Any 20-line LCD
LK_BLINK	Cursor type	2	Steady cursor
		1	Blinking cursor
LG_HDOTS	Horizontal dots	240	TLX-711, TLX-1301
		160	TLX-1013, TLX-1021, TLX-1391
LG_VDOTS	Vertical dots	64	TLX-711, TLX-1021
		128	TLX-1391, TLX-1013, TLX-1301

When using an OP6000 with a controller, remember to include a **#use** call to the controller's Dynamic C library. Refer to the controller manual for which libraries to use.

Initialization Functions

The following **lk_...** functions use the programmable timer PRT1 and may conflict with other software, such as the virtual driver, that uses PRT1.

- **int lk_init_keypad()**

  Performs complete keypad and LCD initialization. Calls **lk_kxinit** and **lk_init**. Starts the keypad interrupt scan timer.

- **int lk_init()**

  Initializes the LCD by clearing its display memory and turning it on.

- **int lk_kxinit()**

  Initializes keypad input by resetting the input buffer, beeper count, and time counter. Call this function at any time to get a "clean slate" for the input buffer.

Character LCD Functions

The following **lk_...** functions use the programmable timer PRT1 and may conflict with other software, such as the virtual driver, that uses PRT1.

- **int lk_tdelay(int delay)**

  Executes a delay expressed in milliseconds.

- **int lk_cgram(char* p)**

  Loads bytes of data into the character generator RAM. The first byte should be the number of remaining bytes to send. Each character is represented by eight bytes, with eight custom characters possible, for a total of 64 bytes maximum. CG RAM addresses start at 40_H and this routine always starts storing at that location. The lower of the data bits (usually the last 5) are actually used to generate the character. Characters in the CG RAM are assigned values 0–7 when writing them.

- **int lk_stdcg()**

  Loads the following eight characters into CG RAM (ASCII equivalents in parentheses): right-arrow (08), down-arrow (09), up-arrow (02), left-arrow (0B), arrow-cursor (0C), up-down-symbol (0D), vertical-line (0E), and horizontal-line (0F).

- **int lk_putc(char x)**

  Sends a single character to the LCD. Will process escape sequences properly but requires multiple calls. This is the equivalent of the standard C function **putc** for the LCD.

- **`int lk_printf( char* fmt, ... )`**

  Works the same as the standard C **printf** statement with output re-directed to the LCD.

- **`int lk_pos( int line, int col )`**

  Moves the cursor to the line and column specified by the given parameters.

LCD Escape Sequences

Escape sequences can be used for special commands to the character LCDs. An escape sequence is a series of characters starting with the escape character (**0x1B**) and followed by control characters. A special function can be initiated by writing these characters to the device. The following is a list of escape characters for the character LCDs.

esc 1	Turn cursor on
esc 0	Turn cursor off
esc c	Erase from current cursor position to end of line.
esc b	Enable blinking cursor mode
esc n	Disable blinking cursor mode
esc e	Erase display and cursor home
esc p n mm	Position cursor to line n and column mm

Examples

```
lk_printf("\x1B1")     // turn on cursor
lk_printf("\x1Bp520") // cursor to row 5, col 20
```

Graphic LCD Functions

The graphic functions provided are for Toshiba displays. Character and graphic operations are treated as if they are on separate displays (see Figure 4-1). It is possible to alternate between the two operations without one affecting the other. An “or/and” or an “exclusive or” display mode can be selected between the two operations.

An X-Y coordinate system is used by graphic functions to locate text or graphic figures on the display. Note that the coordinates differ between the two types of displays. Text is normally written top to bottom and graphics usually require a normal Cartesian coordinate system.

Software routines will correctly address individual pixels only if the correct **`#define`** statements are included in the source program. The default assumes a screen with 240 horizontal pixels and 64 vertical pixels

(used by the TLX-711) and uses an X-Y coordinate system with (0,0) at the lower left corner. The character display uses a 40 × 8 character grid with (0,0) located at the upper left corner. Figure 4-1 illustrates the typical character and graphic coordinate systems.

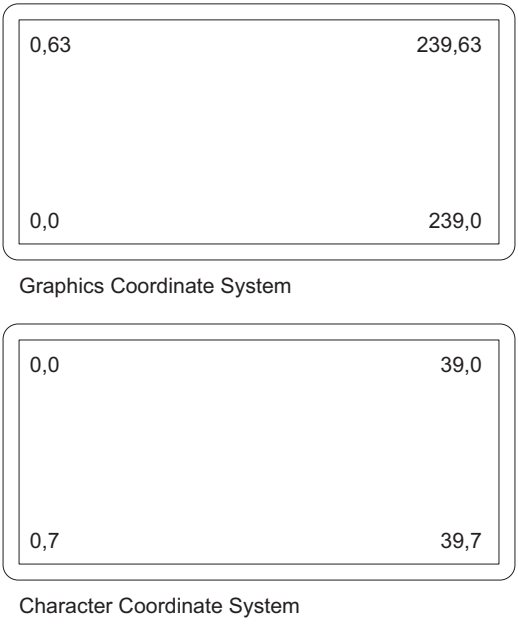


Figure 4-1. Display Coordinate Systems

The graphics write routines all require a mode variable. This parameter tells the routine whether to write, clear, or perform an "exclusive or" mode operation for the function specified. The "exclusive or" operation pertains only to the graphic display. Each bit written is inverted, on to off, or off to on.

Initialization Functions

- **`void Set_Overlap_Mode( int mode )`**

  Sets the write **`mode`** for text and graphics.

`mode`	**meaning**
0	OR mode
1	XOR mode
3	AND mode

- **`void Set_Display_Mode( int dis_mode )`**

  Turns text and graphics modes on or off and handles cursor display (see Figure 4-2). Only the least significant 4 bits of **`dis_mode`** are used.

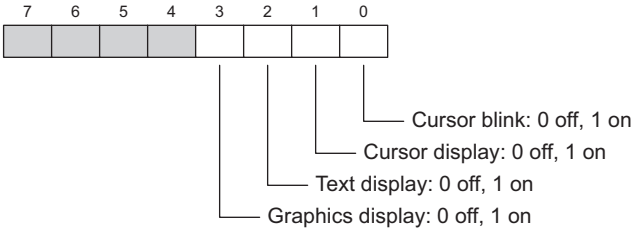


Figure 4-2. Display Mode Bits

Display Functions

- **`void Clear_Gr_Screen()`**

  Erases the graphic display.

- **`int Set_Pixel( int col, int row, int mode )`**

  Writes a pixel at the specified location on the graphics screen. The term **`mode`** has the following values:

mode	**operation**
0	clears pixel
1	sets pixel
2	inverts pixel (XOR)

  The function returns

0	Operation completed
–1	Address boundary error

- **`int Draw_Line( int stx, int sty, int enx, int eny, int mode )`**

  Draws a line from the starting coordinates **`stx`**, **`sty`** to the ending coordinates **`enx`**, **`eny`**. The term **`mode`** has the following values:

mode	**operation**
0	clears pixel
1	sets pixel
2	inverts pixel (XOR)

  The function returns

0	Operation completed
–1	Address boundary error

- **`int Draw_Poly( int n, int* points, int mode )`**

  Draws a (**n**–1)-sided polygon by drawing lines connecting successive points in list. The variables are defined below.

  `n` Number of pairs of X-Y coordinate list **`*points`**. The software does not draw, but moves invisibly, to the first point. The list must have at least two entries (**n** ≥ 2).

  `points` List of alternating X- and Y-coordinates that define the polygon. If you want to close the figure, the last coordinate pair should match the first.

  `mode` write mode

0	clears pixel
1	sets pixel
2	inverts pixel (XOR)

  The function returns

0	operation completed
–1	address boundary error

- **`int Draw_Axis( int ox, int oy, int ex, int ey, int ticks_x, ticks_y, int mode )`**

  Draws a standard X-Y graph axis with tick marks. The variables are defined below.

  `ox, oy` Axis origin coordinate pair.

  `ex` Coordinate at the end of the X-axis. The length of the X-axis is defined as **`ex_ox`**+1.

  `ey` Coordinate of the end of the Y-axis. The length of the Y-axis is defined by **`ey_oy`**+1.

  `ticks_x` Number of ticks to be displayed along the X-axis.

  `ticks_y` Number of ticks to be displayed along the Y-axis.

  `mode` Write mode

0	Clears pixel
1	Sets pixel
2	Inverts pixel (XOR)

  The function returns

0	operation completed
–1	address boundary error

- **`int Map_Bit_Pattern( int* config, char* bitarray, int mode )`**

  Draws a bit-mapped pattern to the graphics screen. The variables are defined below.

  `config` Array of 4 ints that define the bit pattern and its size. The first two values are the XY location of the upper-left corner of the figure. The third int is the width of the figure in pixels. The last int is the height in pixels.

  `bitarray` A character array that defines the bit pattern. A dot is drawn whenever a "1" or "∗" is encountered. This allows drawing figures as arrays in Dynamic C source code. The figure is drawn left-to-right, top-to-bottom.

  `mode` Write mode:

0	Clears pixel
1	Sets pixel
2	Inverts pixel (XOR)

  The function returns:

0	Operation completed
–1	Address boundary error

Example

Draw arrowhead at location (100,40).

```
char arrowhead[6][7] = { "   *   "
                         "  ***  "
                         " ***** "
                         "*******"
                         "  ***  "
                         "  ***  "  };

int arrowdef = {100,40,6,7};

main(){
   ...
  Map_Bit_Pattern( arrowdef, arrowhead, 1 );
   ...

}
```

Character Functions

- **`void Define_Cursor( int lines )`**

  Defines the cursor from 1 to 8 lines in height.

- **`void Clear_Txt_Screen()`**

  Erases the text display.

The following **`lg_...`** functions use the programmable timer PRT1 and may conflict with other software, such as the virtual driver, that uses PRT1.

- **`int lg_putc( char x )`**

  Sends a single character to the character memory of the graphics LCD. Processes escape sequences properly under multiple calls. This is the equivalent of the standard C function **`putc`**, but for the LCD.

- **`int lg_printf( char* fmt, ... )`**

  Works the same as the standard C **`printf`** function, but with output directed to the graphics LCD character memory.

- **`int lg_pos( int line, int col )`**

  Moves the cursor to the line and column specified by the given parameters.

Keypad Functions

The following **`lk_...`** function uses the programmable timer PRT1 and may conflict with other software, such as the virtual driver, that uses PRT1.

- **`int lk_kxget()`**

  Performs a keypad scan and returns the key number if one is pressed. Returns –1 if no key is pressed at the time. See the keypad layout and termination schedules in Figures 3-2 and 3-3 and Tables 3-1 and 3-2 for key numbers.

Beeper Functions

The following **`lk_...`** function uses the programmable timer PRT1 and may conflict with other software, such as the virtual driver, that uses PRT1.

- **`int lk_setbeep( int count )`**

  Activates the beeper for **`count`** milliseconds.

Blank

G

H

I

J

K

L

M

O

P

R

S

T

Blank

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

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

Part No. 019-0020
Revision D

Printed in U.S.A.