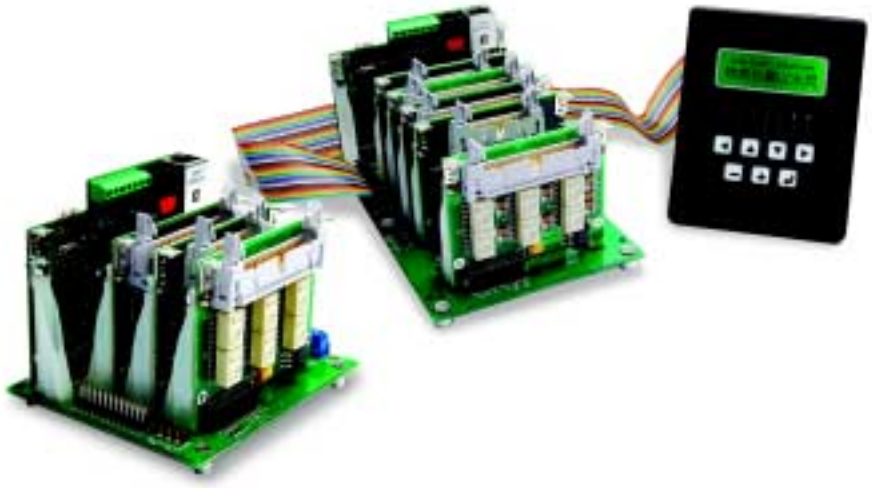


Smart Star (SR9000)

Modular C-Programmable Control System

User's Manual

019–0107 • 020301–A

Smart Star (SR9000) User's Manual

Part Number 019-0107 • 020301 - A • 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.

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.

Trademarks

Rabbit 2000 is a trademark of Rabbit Semiconductor.

Dynamic C is a registered trademark of Z-World Inc.

Z-World, Inc.

2900 Spafford Street
Davis, California 95616-6800
USA

Telephone: (530) 757-3737
Fax: (530) 753-5141

www.zworld.com

TABLE OF CONTENTS

PART I. CPU/Backplane

PART IV. D/A Converter Cards

PART V. Relay Cards

PART VI. Appendices

PART I. CPU/BACKPLANE

1. INTRODUCTION

Chapter 1 introduces the Smart Star embedded control system and describes the features associated with the backplane chassis and the CPU card. The Tool Kit containing the hardware essentials to begin using the Smart Star is described, and the software highlights are presented.

The Smart Star is a modular and expandable embedded control system whose configuration of digital I/O, A/D converter, D/A converter, and relay I/O cards can be tailored to a large variety of demanding real-time control and data acquisition applications.

The typical Smart Star system consists of a rugged backplane with a built-in voltage regulator, a CPU card, and one or more I/O cards. The CPU card plugs into a designated slot on the backplane chassis, which has additional slots available for any combination of I/O cards. A high-performance Rabbit 2000 microprocessor on the CPU card operates at 22.1 MHz to provide fast data processing.

1.1 Features

- C-programmable to create a custom user interface
- Flexible functionality—modular configuration allows interchanging or replacing individual I/O cards
- Expandable—up to 168 I/O ports
- Choice of two backplanes—with either 3 or 7 slots for I/O cards
- Choice of CPU cards—with or without One RJ-45 Ethernet port compliant with IEEE 802.3 standard for 10Base-T Ethernet protocol
- RS-232 and RS-485 serial ports allow networking to other Smart Star units, single-board computers, or enterprise computing centers
- 128K SRAM and 512K flash memory, optional 512K SRAM
- Real-time clock
- Watchdog supervisor
- Backup battery
- Optional backlit 122 × 32 graphic display/keypad module
- RabbitLink Ethernet gateway available for remote download/debug, Web serving, and e-mail

Table 1 lists the backplanes, CPU cards, and the I/O cards that are available for the Smart Star control system. Appendix A provides detailed specifications for the Smart Star backplanes and the CPU cards.

Table 1. Smart Star Backplanes and Cards

Card		Model	Features
Backplane		SR9010	7 I/O card slots, 1 CPU card slot
		SR9050	3 I/O card slots, 1 CPU card slot, header connections for optional LCD/keypad module
CPU		SR9150	Full-featured CPU card ***with*** RJ-45 Ethernet port
		SR9160	Full-featured CPU card ***without*** RJ-45 Ethernet port
I/O Cards	Digital I/O	SR9200	16 digital inputs, 8 digital sinking outputs
		SR9210	8 digital inputs, 16 digital sinking outputs
		SR9220	8 digital inputs, 8 digital sinking outputs
		SR9205	16 digital inputs, 8 digital sourcing outputs
		SR9215	8 digital inputs, 16 digital sourcing outputs
		SR9225	8 digital inputs, 8 digital sourcing outputs
	A/D Converter	SR9300	12-bit A/D converter, 11 channels, 0 V – 10 V
		SR9310	12-bit A/D converter, 11 channels, -10 V – +10 V
		SR9320	12-bit A/D converter, 11 channels, 4 mA – 20 mA
	D/A Converter	SR9400	12-bit D/A converter, 8 channels, 0 V – 10 V
		SR9410	12-bit D/A converter, 8 channels, -10 V – +10 V
		SR9420	12-bit D/A converter, 8 channels, 4 mA – 20 mA
	Relay	SR9500	5 SPST relays and 1 SPDT relay, each protected with onboard snubbers
		SR9510	8 SPDT relays (no snubbers)

1.2 User Connections

Connections to the I/O cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Three different Field Wiring Terminals (FWTs) are available. Table 2 lists the I/O cards and the Z-World part numbers for the corresponding FWTs.

Table 2. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals 	Screw Terminals 
FWT27	Digital I/O Relay (SR9510)	101-0420	101-0424
FWT18	A/D Converter D/A Converter	101-0421	101-0425
FWT18R	Relay (SR9500)	101-0422	101-0426

NOTE: Appendix A, "Field Wiring Terminals," provides further information on FWTs, including their dimensions.

1.3 Optional Add-Ons

The LCD/keypad module is the only available optional add-on. Further details on the LCD/keypad module are provided in Appendix B.

Visit Z-World's Web site for up-to-date information about additional add-ons and features as they become available. The Web site also has the latest revision of this user's manual.

1.4 Development and Evaluation Tools

1.4.1 Tool Kit

The Tool Kit has the hardware essentials that you need to create and use your own Smart Star control system.

The items in the Tool Kit and their use are as follows:

- ***Smart Star (SR9000) User's Manual*** with schematics for the backplane chassis and the CPU cards (this document).
- User manuals for available I/O cards.
- Programming cable, used to connect your PC serial port to the Smart Star CPU card to write and debug C programs that run on the Smart Star control system.
- FWT27 pluggable field wiring terminal.
- Screwdriver.
- DC power supply, used to power the backplane, which in turn supplies power to the CPU card and the I/O cards. The DC power supply accepts an AC input of 100 V to 240 V at up to 0.6 A, and delivers a DC output up to 1.1 A at 24 V.
- ***Rabbit 2000 Processor Easy Reference*** poster.
- Registration card.

1.4.2 Software

The Smart Star control system is programmed using Z-World's Dynamic C Premier, an integrated development environment that includes an editor, a C compiler, and a debugger. Library functions and software drivers provide an easy-to-use interface for the Smart Star control system. Dynamic C Premier is sold separately.

2. GETTING STARTED

Chapter 2 explains how to connect the power supply to the Smart Star backplane, how to install the CPU card on the backplane, and how to connect the programming cable to the CPU card. Once you run a sample program to demonstrate that you have connected everything correctly, you will be ready to go on to install I/O cards and finish developing your system.

2.1 Attach the CPU Card to the Backplane

1. Orient the backplane with the PROCESSOR SLOT facing away from you as shown in Figure 1.

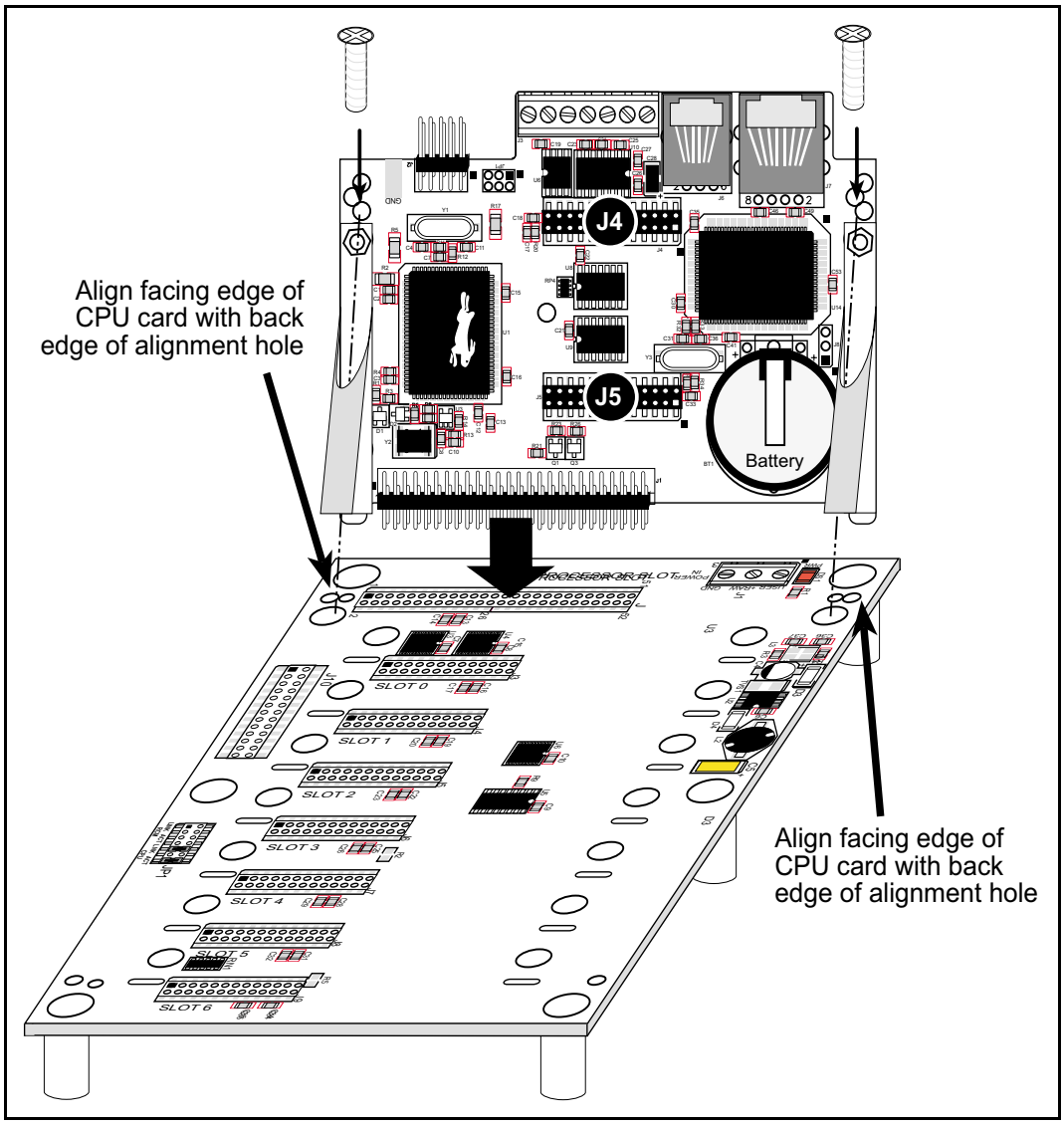


Figure 1. Attach the CPU Card to the Backplane

2. Position the CPU card above the backplane as shown in Figure 1.
3. Carefully insert the CPU card header into the **PROCESSOR SLOT** on the backplane and line up the facing edge of the CPU card with the back edge of the alignment holes on the backplane as shown in Figure 1.

   NOTE: Be careful to line up the pins on the CPU card with the socket on the backplane when installing the CPU card. The CPU card can be damaged once power is applied if the CPU card is not installed correctly.

4. Use the two 4-40 screws supplied with the CPU card to anchor the plastic brackets so that they hold the CPU card firmly in place on the backplane.

2.2 Connect the Power Supply

Connect the power supply to the **POWER IN** connector on the backplane—the red (positive) wire to **+RAW** and the black (negative) wire to **GND**, as shown in Figure 2.

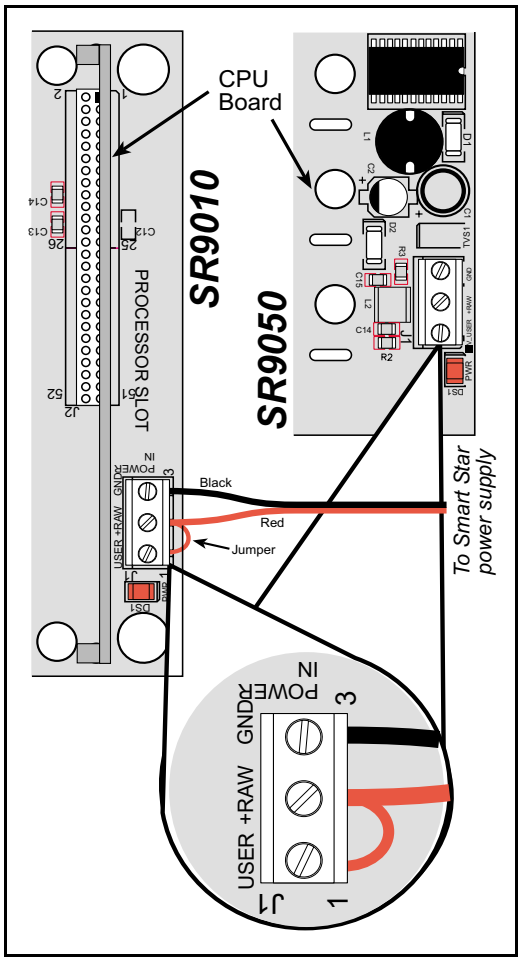


Figure 2. Power Supply Connections

> **NOTE:** Be careful to hook up the positive and negative leads ***exactly*** as described. Only the +5 V circuitry is protected against reverse polarity.

A **USER** connection is supplied on the backplane to allow an independent power supply to be used for future development. For now, use a jumper to connect **USER** to **+RAW** so that they share the same power supply.

Notice to Customers Outside North America

The power supply included with the Smart Star Tool Kit may be used worldwide. Customers outside North America simply need to exchange the line cord and plug from the power supply to their wall outlet with one available locally.

1. To exchange the line cord and plug, first remove the existing line cord. To access the screws, use a screwdriver to gently lift up and remove the plastic insulating cover.

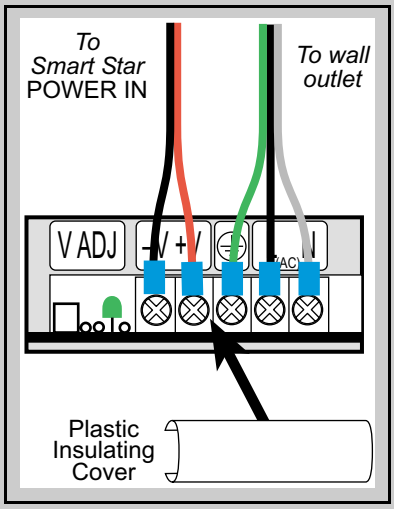


Figure 3. Power Supply Connections

2. Unscrew the wires at the ground, **L**, and **N** terminals.
3. Attach the line cord that you obtained locally to the power supply. Be sure to follow any color-coding conventions, for example, green/yellow to ground, brown to **L**, and blue to **N** terminals.
4. Ensure that the wires are attached securely and are not touching each other. Snap on the plastc insulating cover.

> **NOTE:** The power supply included with the Smart Star Tool Kit is intended for development purposes only.

2.3 Programming Cable Connections

1. Connect the programming cable to the CPU card.

Connect the 10-pin **PROG** connector of the programming cable to header J2 on the CPU card as shown in Figure 4. Connect the other end of the programming cable to a COM port on your PC. Note that COM1 on the PC is the default COM port in the Dynamic C Premier installation.

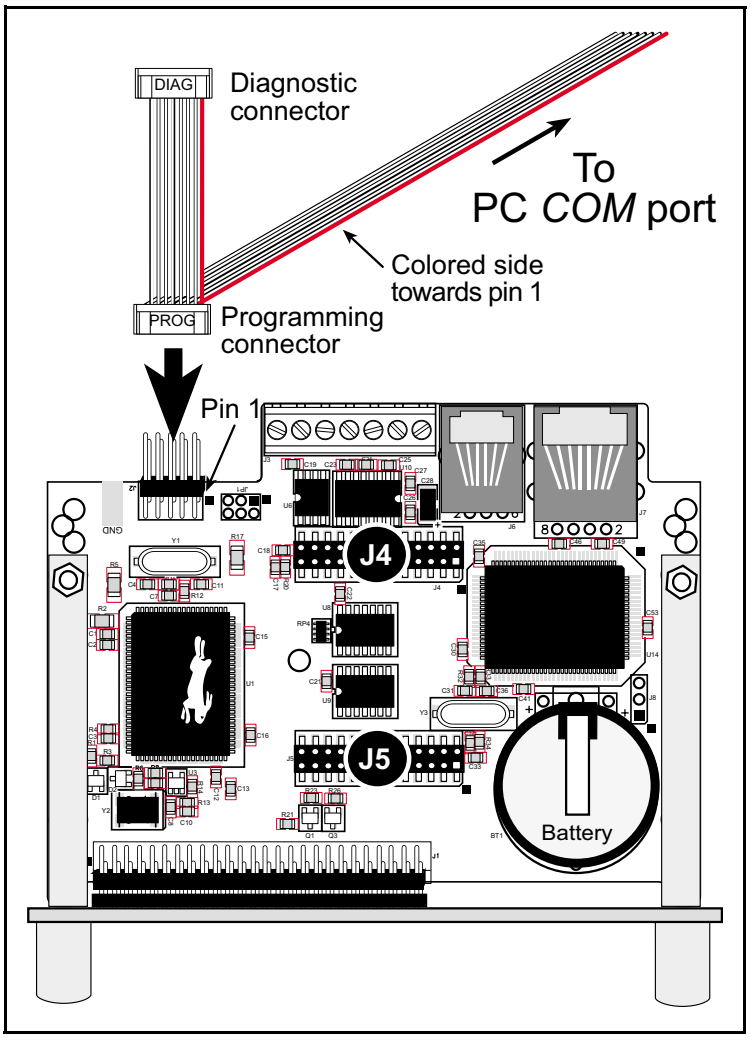


Figure 4. Programming Cable Connections

2. **Apply power.**

Plug the power supply in to a nearby outlet. The CPU card is now ready to be used.

NOTE: A hardware RESET is accomplished by unplugging the power supply, then plugging it back in.

2.4 Installing Dynamic C Premier

If you have not yet installed Dynamic C version 7.06P3 (or a later version), do so now by inserting the Dynamic C Premier CD in your PC's CD-ROM drive. The CD will auto-install unless you have disabled auto-install on your PC.

If the CD does not auto-install, click **Start > Run** from the Windows **Start** button and browse for the **setup.exe** file on your CD drive. Click **OK** to begin the installation once you have selected the **setup.exe** file.

The ***Dynamic C Premier User's Manual*** provides detailed instructions for the installation of Dynamic C and any future upgrades.

> **NOTE:** If you have an earlier version of Dynamic C already installed, the default installation of the later version will be in a different folder, and a separate icon will appear on your desktop.

2.5 Starting Dynamic C

Once the CPU card is installed and connected as described above, start Dynamic C by double-clicking on the Dynamic C icon or by double-clicking on the **.exe** file associated with **DcRab** in the Dynamic C directory.

Dynamic C should detect the CPU card and go through a sequence of steps to cold-boot the CPU card and to compile the BIOS. If the error message "Rabbit Processor Not Detected" appears, you have probably connected to a different PC serial port than the default serial port selected during the installation of Dynamic C Premier. You can change the serial port used by Dynamic C with the **OPTIONS** menu, then try to get Dynamic C to recognize the CPU card by selecting **Recompile BIOS** on the **Compile** menu. Try the different COM ports in the **OPTIONS** menu until you find the one you are connected to. If you still cannot get Dynamic C to recognize the target on any port, then the hookup may be wrong or the COM port is not working on your PC.

If you receive the "BIOS successfully compiled …" message after pressing **<Ctrl-Y>** or starting Dynamic C, and this message is followed by a communications error message, it is possible that your PC cannot handle the 115,200 bps baud rate. Try changing the baud rate to 57,600 bps as follows.

- Locate the **Serial Options** dialog in the Dynamic C **Options > Communications** menu. Change the baud rate to 57,600 bps.

2.6 PONG.C

You are now ready to test your set-up by running a sample program.

Find the file **PONG.C**, which is in the Dynamic C **SAMPLES** folder. To run the program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The **STDIO** window will open and will display a small square bouncing around in a box.

This program does not test the serial ports on the CPU card, but does ensure that the CPU is basically functional.

2.7 Installing I/O Cards

1. Orient the backplane with the CPU card already installed and facing towards you as shown in Figure 5.

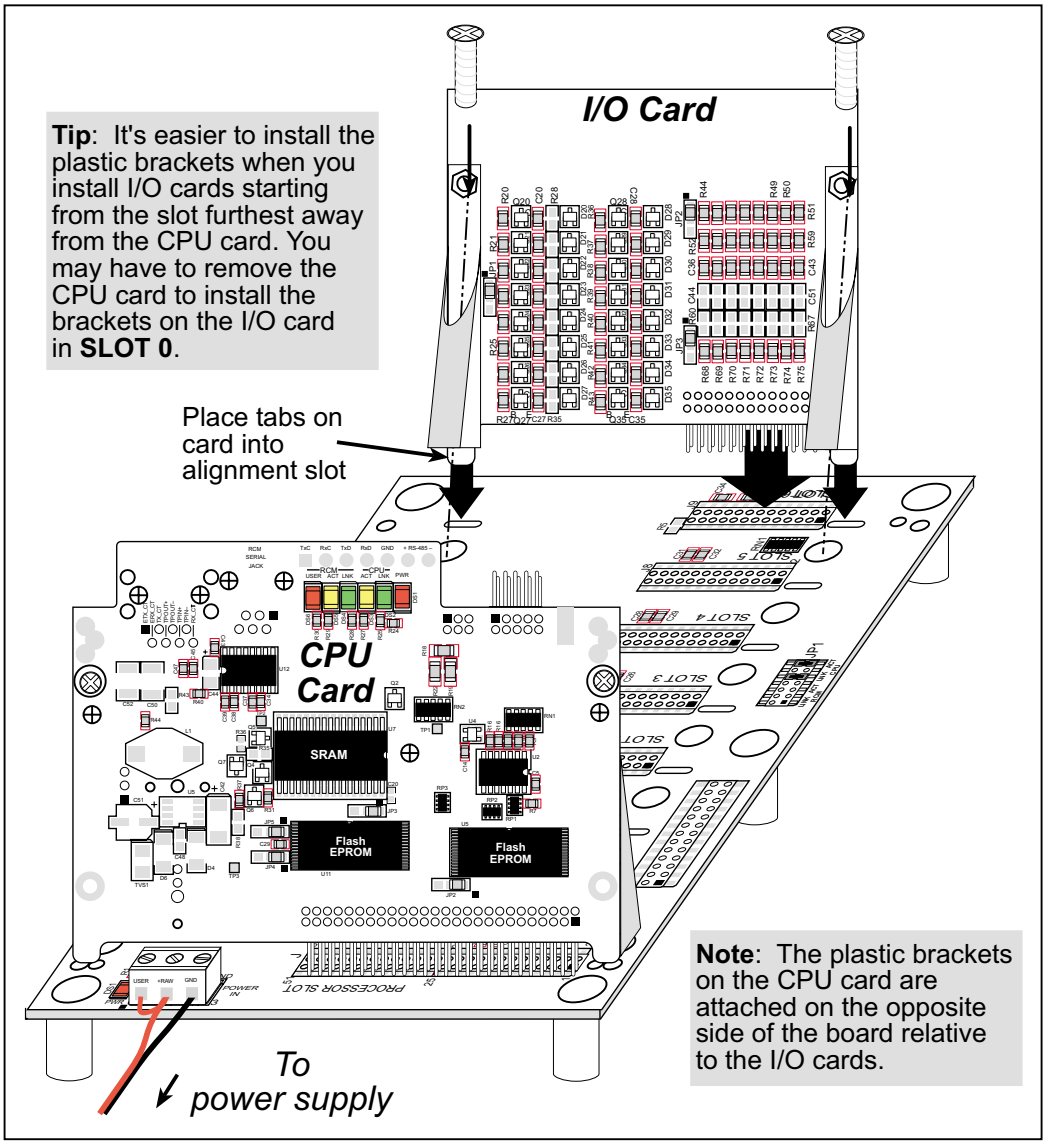


Figure 5. Installing I/O Cards on the Backplane

2. Position the new I/O card above the backplane over any unused slot position (**SLOT 0** to **SLOT 6**) as shown in Figure 5. Note the slot number and the type of I/O card since Dynamic C addresses the I/O cards by slot number.
3. Carefully insert the I/O card header into the slot on the backplane and line up the tabs on the I/O cards with the slots on the backplane as shown in Figure 5.
4. Use the two 4-40 screws supplied with the I/O card to anchor the plastic brackets on the CPU card or the I/O card firmly on the backplane. Tighten the screws as needed using a Phillips screwdriver whose shaft is at least 3" (7 cm) long, but is no thicker than 0.16" (4 mm).

2.8 Where Do I Go From Here?

NOTE: If you purchased your Smart Star through a distributor or Z-World partner, contact the distributor or Z-World partner first for technical support.

If there are any problems at this point:

- Check the Z-World Technical Bulletin Board at www.zworld.com/support/bb/.
- Use the Technical Support e-mail form at www.zworld.com/support/support_submit.html.
- Call Z-World Technical Support at (530)757-3737.

If the sample program ran fine, you are now ready to go on to install I/O cards, explore other Smart Star features, and develop your own applications.

Chapter 3, "Hardware Features," provides detailed information about the CPU card, and how to install the I/O cards. Be sure to take the total current consumption of the individual cards into account when selecting a power supply. Appendix C.1, "Current Requirements," provides more detailed information. Chapter 4, "Software," describes the Dynamic C software libraries and introduces some sample programs for use with the CPU card. Chapter 6, "Smart Star Specifications," provides specifications for the backplanes and the CPU cards, including mounting and clearance recommendations.

Separate sections in this manual have been prepared for the various I/O cards, and include complete information about their pinouts and Dynamic C software libraries, including sample programs.

Once you have developed your application and bench-tested the finished system, you may install the finished system.

3. HARDWARE FEATURES

Chapter 3 describes the principal features for the Smart Star backplanes and CPU cards.

- Power Distribution
  - Power Distribution
  - I/O Card Slots
- Smart Star CPU Card Features
  - Serial Communication
  - Memory
  - Other Connectors

3.1 Backplane Features

3.1.1 Power Distribution

Power is supplied to the Smart Star control system from an external source through header J1 on the backplane. The +5 V circuitry on the Smart Star control system is protected against reverse polarity by a Schottky diode as shown in Figure 6.

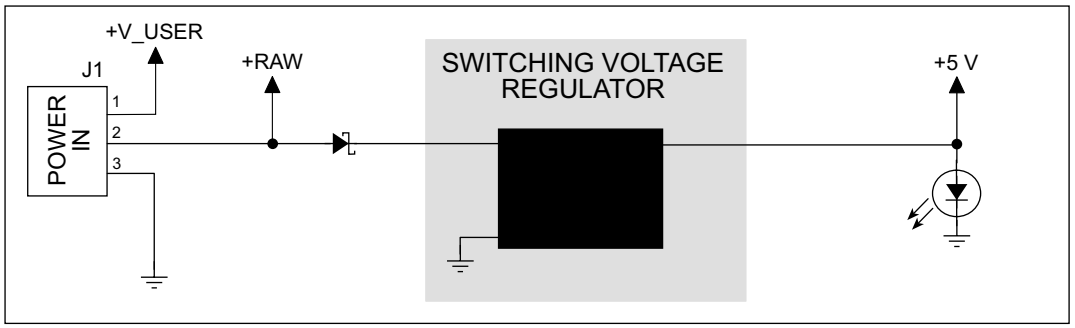


Figure 6. Smart Star Control System Power Supply Schematic

A capacitor provides surge current protection for the voltage regulator, and allows the external power supply to be located some distance away from the Smart Star control system. A switching power regulator is used. The **+RAW** input voltage may range from 9 V to 30 V (15 V to 30 V you plan to use a D/A converter card).

The backplane has inputs for two separate power supplies on header J1, **+RAW** and **V_USER**. The **+RAW** power supply goes to the switching power regulator, which outputs the +5 V DC used by the CPU card and by the I/O cards plugged into the backplane. The **V_USER** connection allows a different voltage to be available on the I/O cards for future development.

> **NOTE:** Always connect **V_USER** to **+RAW** with a jumper wire between terminals 1 and 2 on header J1 for the development activities described in the Smart Star manuals.

Figure 7 shows how the power supplies are distributed on the backplane and on the CPU card.

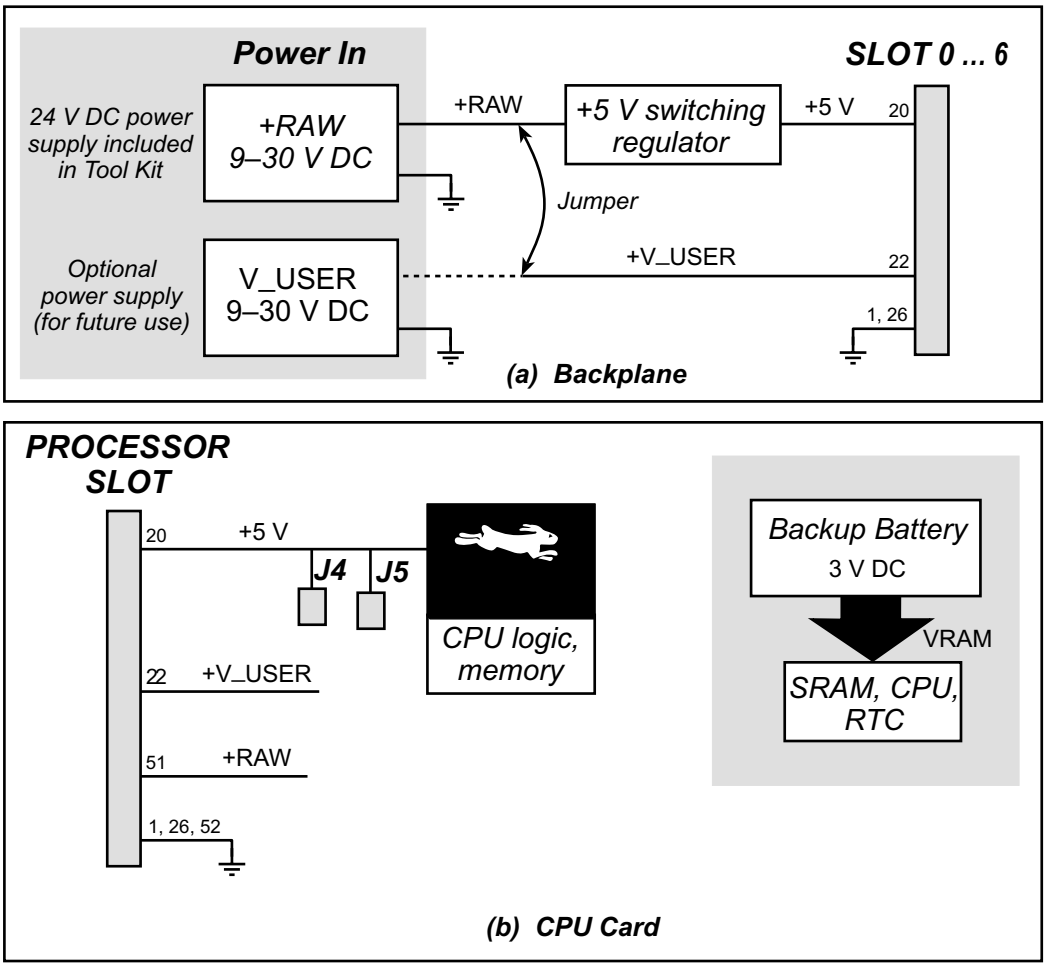


Figure 7. Smart Star Power Supplies—Backplane and CPU Card

Figure 8 shows how the power supplies are distributed on the I/O cards.

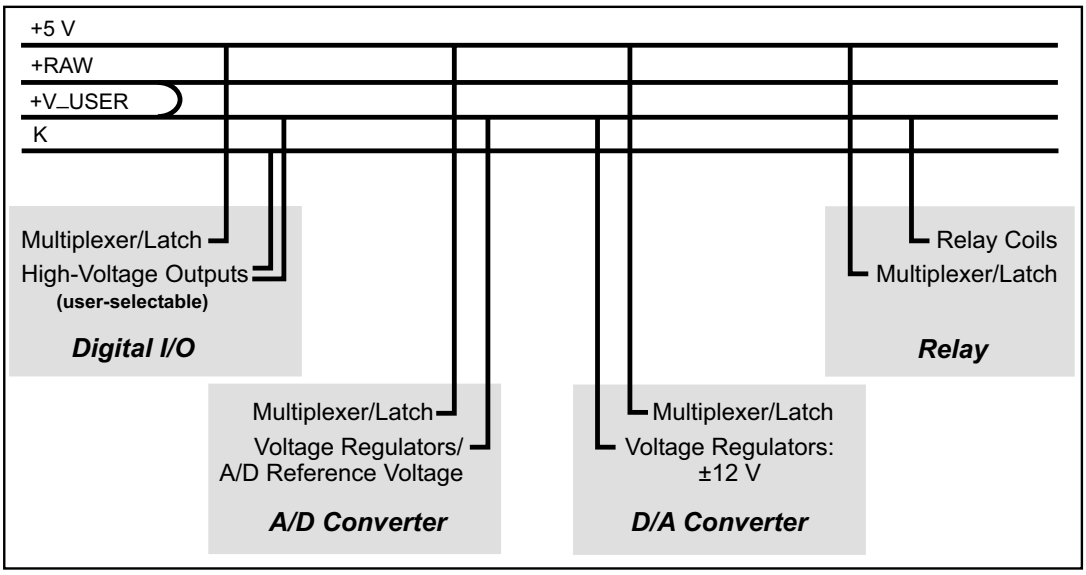


Figure 8. Smart Star Power Distribution on I/O Cards

NOTE: Note that Z-World recommends tying **+RAW** to **+V_USER** as explained in Section 2.2, "Connect the Power Supply."

The user has the option of using a separate power supply to K when configuring the high-power outputs for the digital I/O cards. The connection to K is through the user interface on the digital I/O card. Further details are provided in Chapter 7, "Digital I/O Cards."

3.1.2 I/O Card Slots

The backplane serves to make the CPU card accessible to up to seven I/O cards plugged in to **SLOT 0** through **SLOT 6** on the backplane. Figure 9 shows the pinout for **SLOT 0** through **SLOT 6** (headers J3–J9) on the backplane.

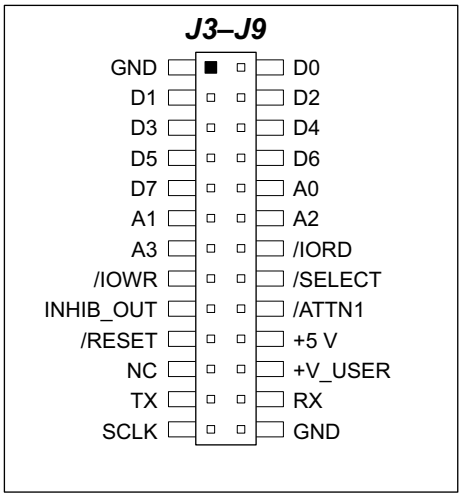


Figure 9. Pinout for SLOT 0 Through SLOT 6 (Headers J3–J9) on the Backplane

NOTE: The SR9050 backplane can accomodate up to three I/O cards plugged in to **SLOT 0** through **SLOT 2** (headers J3–J5).

3.2 Smart Star CPU Card Features

3.2.1 Serial Communication

The CPU card has one screw terminal header for RS-232/RS-485 serial communication (J3) and one RJ-45 Ethernet jack (J7, SR9150 only). The RJ-12 jack, J6, is reserved for future use and therefore has no signals. The pinouts are shown in Figure 10.

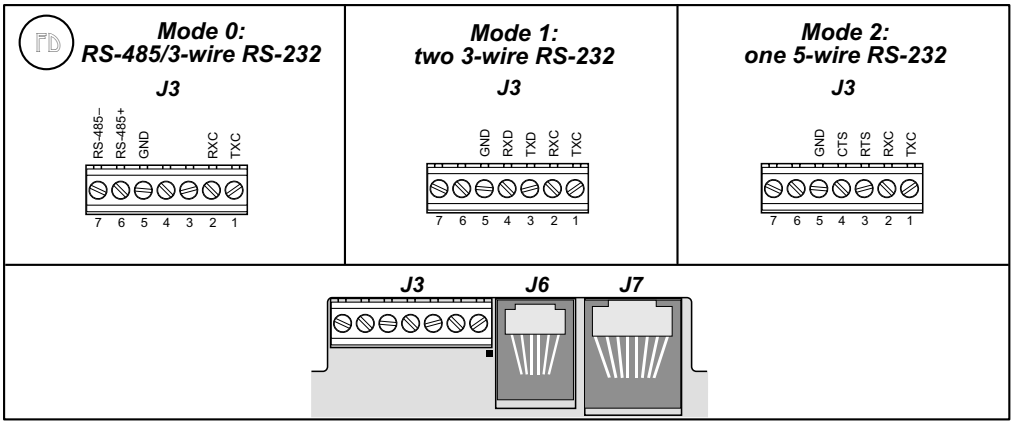


Figure 10. Smart Star CPU Card Serial Pinout

The factory default for the CPU card is one RS-232 (3-wire) and one RS-485 serial channel, corresponding to Mode 0 in Figure 10. The other modes shown in Figure 10 are set in software via the Dynamic C **`serMode`** function call (see Section 4.4, "Serial Communication APIs").

3.2.1.1 RS-232

The CPU card's RS-232 serial channel is connected to an RS-232 transceiver. The transceiver provides the voltage output, slew rate, and input voltage immunity required to meet the RS-232 serial communication protocol. Basically, the chip translates the Rabbit 2000's 0 V to +Vcc signals to RS-232 signal levels. Note that the polarity is reversed in an RS-232 circuit so that +5 V is output as approximately -10 V and 0 V is output as approximately +10 V. The transceiver also provides the proper line loading for reliable communication.

The maximum baud rate is 115,200 bps. RS-232 can be used effectively at this baud rate for distances up to 15 m.

The Rabbit 2000 serial port C TXD and RXD signals are presented either as RS-232 TX and RX or as RTS/CTS handshaking, depending on the mode selected with the Dynamic C function **`serMode`**. The RS-232 signals are available on screw terminal header J3.

3.2.1.2 RS-485

The CPU card has one RS-485 serial channel, which is connected to the Rabbit 2000 serial port C through an RS-485 transceiver. The chip's slew rate limiters provide for a maximum baud rate of 250,000 bps, and allows networking over a distance of up to 300 m (or 1000 ft.). The half-duplex communication uses the Rabbit 2000's PD4 pin to control the data enable on the communication line.

The RS-485 signals are available on the CPU card through screw terminal header J3.

The Smart Star control system can be used in an RS-485 multidrop network. Connect the 485+ to 485+ and 485– to 485– using single twisted-pair wires on the CPU card's header J4 as shown in Figure 11. Note that a common ground is recommended.

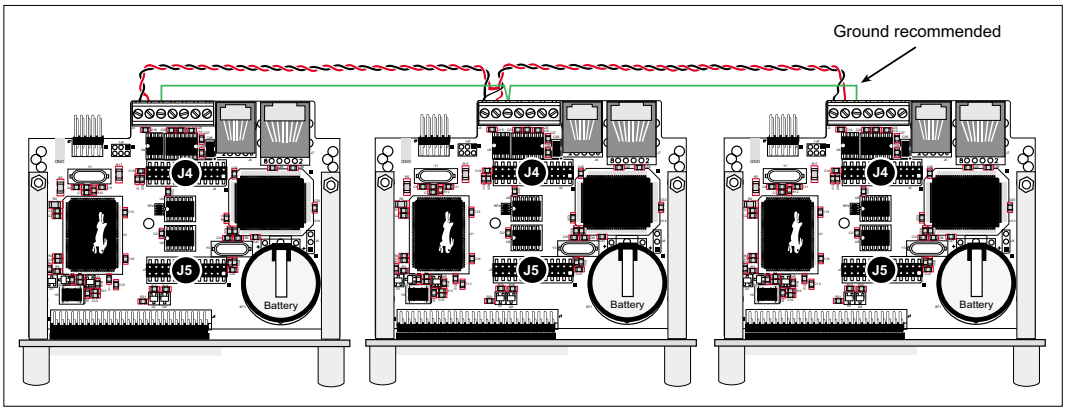


Figure 11. Multidrop Smart Star Network

The CPU card comes with a 220 Ω termination resistor and 681 Ω bias resistors already installed and enabled with jumpers across pins 1–2 and 5–6 on header JP1, as shown in Figure 12.

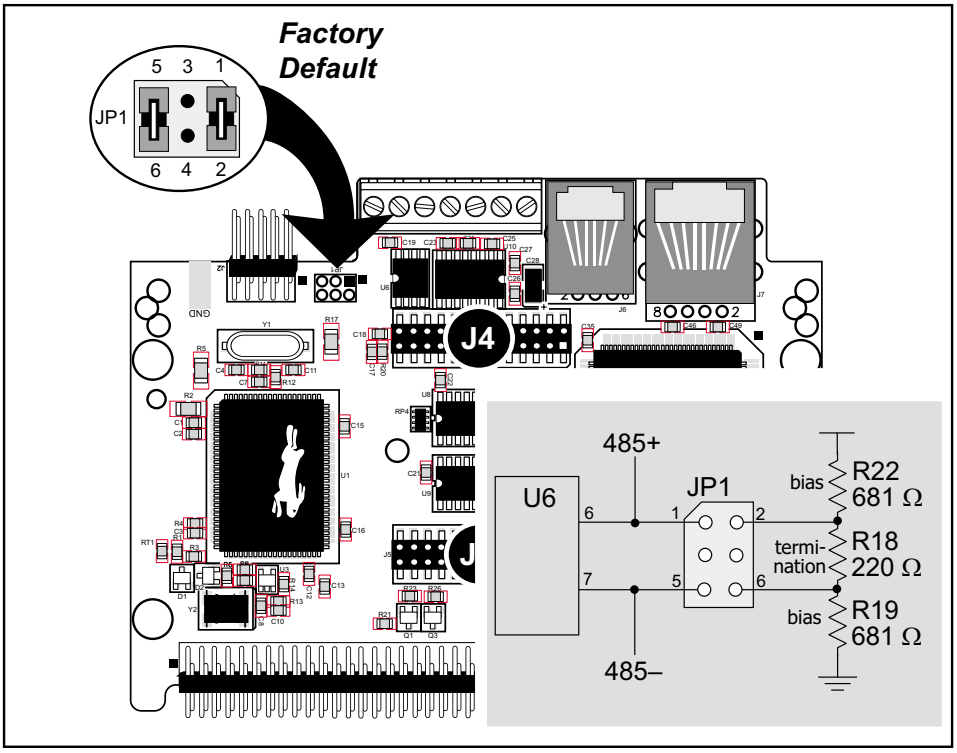


Figure 12. RS-485 Termination and Bias Resistors

For best performance, the bias and termination resistors in a multidrop network should only be enabled on both end nodes of the network. Disable the termination and bias resistors on any intervening Smart Star units in the network by removing both jumpers from header JP1.

> **TIP:** Save the jumpers for possible future use by "parking" them across pins 1–3 and 4–6 of header JP1. Pins 3 and 4 are not otherwise connected to the CPU card.

3.2.1.3 Programming Port

The CPU card has a 10-pin programming header labeled J2. The programming port uses the Rabbit 2000's Serial Port A for communication, and is used for the following operations.

- Programming/debugging
- Cloning
- Remote program download/debug over an Ethernet connection via the RabbitLink EG2100

The programming port is used to start the Smart Star in a mode where the CPU card will download a program from the port and then execute the program. The programming port transmits information to and from a PC while a program is being debugged.

The Rabbit 2000 startup-mode pins (SMODE0, SMODE1) are presented to the programming port so that an externally connected device can force the Rabbit 2000 to start up in an external bootstrap mode.

> **NOTE:** Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information related to the bootstrap mode.

The programming port can also be used as a diagnostic port when the **DIAG** connector on the programming cable is used to connect the programming port to a PC or other device. See Appendix D, "Programming Cable," for more information.

3.2.1.4 **Ethernet Port** (SR9150 only)

Figure 13 shows the pinout for the Ethernet port (J2 on the CPU card). Note that there are two standards for numbering the pins on this connector—the convention used here, and numbering in reverse to that shown. Regardless of the numbering convention followed, the pin positions relative to the spring tab position (located at the bottom of the RJ-45 jack in Figure 13) are always absolute, and the RJ-45 connector will work properly with off-the-shelf Ethernet cables.

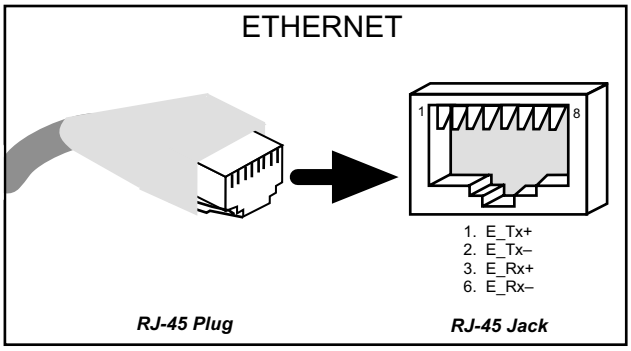


Figure 13. RJ-45 Ethernet Port Pinout

RJ-45 pinouts are sometimes numbered opposite to the way shown in Figure 13.

Two LEDs are placed behind the RJ-45 Ethernet jack, one to indicate an Ethernet link (**LNK**) and one to indicate Ethernet activity (**ACT**). Only the CPU LEDs are functional at this time since the RCM LEDs were added for future enhancements to the CPU card.

The transformer/connector assembly ground is connected to the CPU card digital ground via a 0 Ω resistor "jumper," R43, as shown in Figure 14.

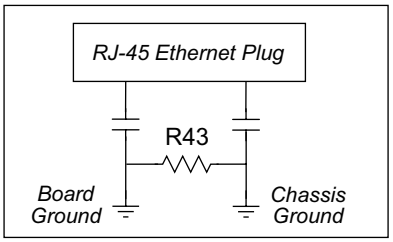


Figure 14. Isolation Resistor R43

The factory default is for the 0 Ω resistor "jumper" at R43 to be installed. In high-noise environments, remove R43 and ground the transformer/connector assembly directly through the chassis ground. This will be especially helpful to minimize ESD and/or EMI problems.

3.2.2 Memory

3.2.2.1 SRAM

The Smart Star CPU cards are designed to accept 128K or 512K of static RAM packaged in an SOIC case. Standard CPU cards come with 128K of SRAM.

3.2.2.2 Flash EPROM

The Smart Star CPU card are also designed to accept 128K to a total of 512K of flash memory packaged in a TSOP case. The CPU cards come with two 256K flash memory chips.

> **NOTE:** Z-World recommends that any customer applications should not be constrained by the sector size of the flash memory since it may be necessary to change the sector size in the future.

A Flash Memory Bank Select jumper configuration option based on 0 Ω surface-mounted resistors exists at header JP5 on the CPU card. This option, used in conjunction with some configuration macros, allows Dynamic C to compile two different co-resident programs for the upper and lower halves of the 256K flash in such a way that both programs start at logical address 0000. This is useful for applications that require a resident download manager and a separate downloaded program. See Technical Note 218, ***Implementing a Serial Download Manager for a 256K Flash***, for details.

3.2.3 Other Connectors

The connectors labeled J4 and J5 in Figure 15 are reserved for future use and should not be used in customer applications at this time.

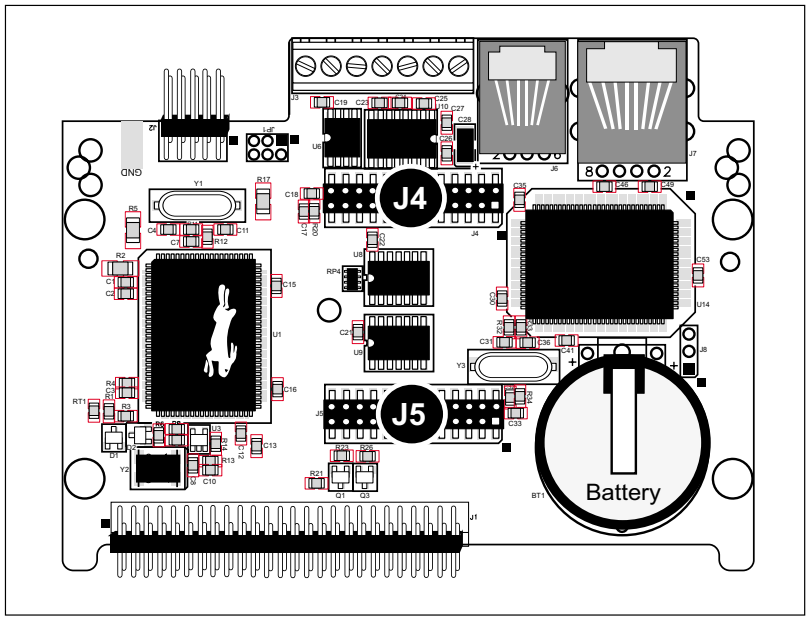


Figure 15. CPU Card Connectors J4 and J5

Jumpers across pins 9–10 and 13–14 on header JP1 on the backplane are used to bring out the **ACT** and **LNK** LED signals to header J6, which is used to connect the optional LCD/keypad module. Remove these jumpers (you may park them across pins 7–8 and 11–12 on header JP1) if you do not wish to use the **ACT** and **LNK** signals on the LCD/keypad module.

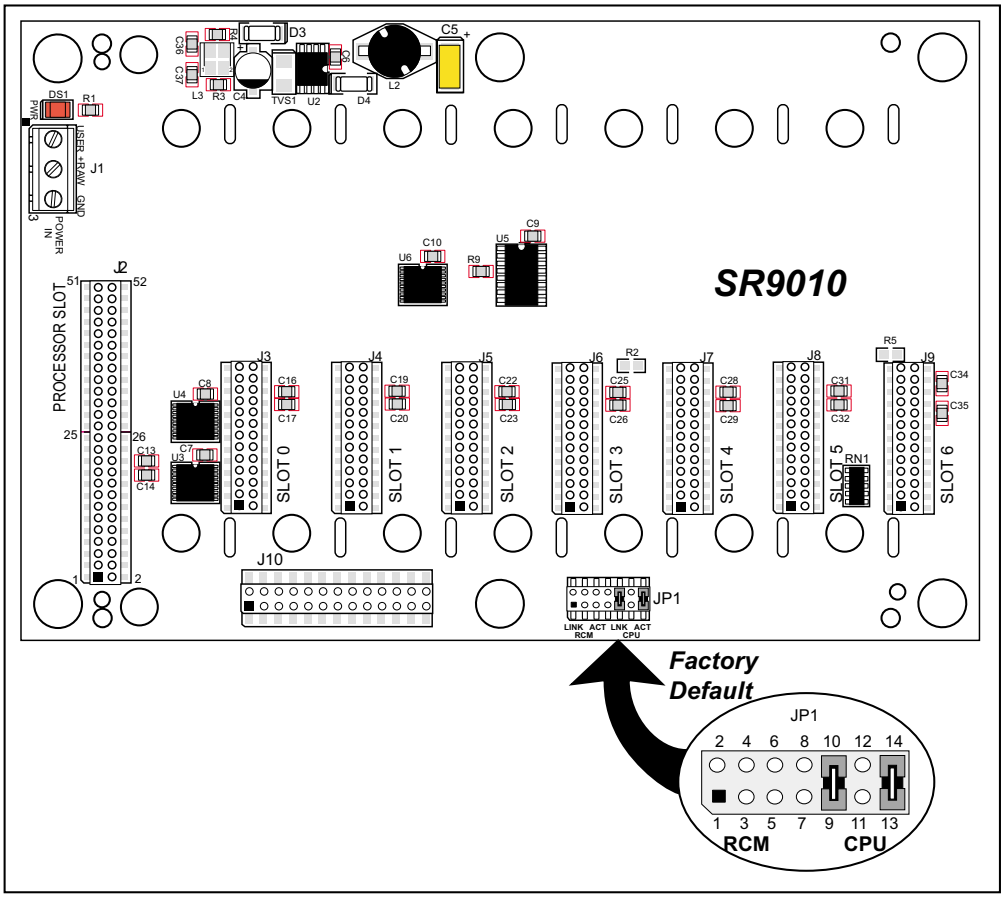


Figure 16. Header JP1 Configurations for* ACT *and* LNK *Signals

NOTE: The RCM positions for pins 1–2 and 5–6 on header JP1 are reserved for future use and should not be used in customer applications at this time.

4. SOFTWARE

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

Chapter 4 provides the libraries, function calls, and sample programs related to the Smart Star backplane and CPU cards.

You have a choice of doing your software development in the flash memory or in the static RAM included on the Smart Star CPU cards. The advantage of working in RAM is to save wear on the flash memory, which is limited to about 100,000 write cycles.

NOTE: An application can be developed in RAM, but cannot run standalone from RAM after the programming cable is disconnected. All standalone applications can only run from flash memory.

The disadvantage of using flash memory for debug is that interrupts must be disabled for approximately 5 ms whenever a break point is set in the program. This can crash fast interrupt routines that are running while you stop at a break point or single-step the program. Flash memory or RAM is selected on the **Options** > **Compiler** menu.

Dynamic C Premier provides a number of debugging features. You can single-step your program, either in C, statement by statement, or in assembly language, instruction by instruction. You can set break points, where the program will stop, on any statement. You can evaluate watch expressions. A watch expression is any C expression that can be evaluated in the context of the program. If the program is at a break point, a watch expression can view any expression using local or external variables.

4.1 Programming Cable

The programming cable has a level converter board in the middle of the cable since the programming port on the CPU card supports CMOS logic levels, and not the higher voltage RS-232 levels that are used by PC serial ports. When the programming cable is connected, Dynamic C running on the PC can hard-reset the Smart Star and cold-boot it. The cold boot includes compiling and downloading a BIOS program that stays resident while you work. If you crash the target, Dynamic C will automatically reboot and recompile the BIOS if it senses that a target communication error occurred or that the BIOS source code has changed.

4.1.1 Switching Between Program Mode and Run Mode

The CPU card is automatically in Program Mode when the programming cable is attached, and is automatically in Run Mode when no programming cable is attached. See Figure 17.

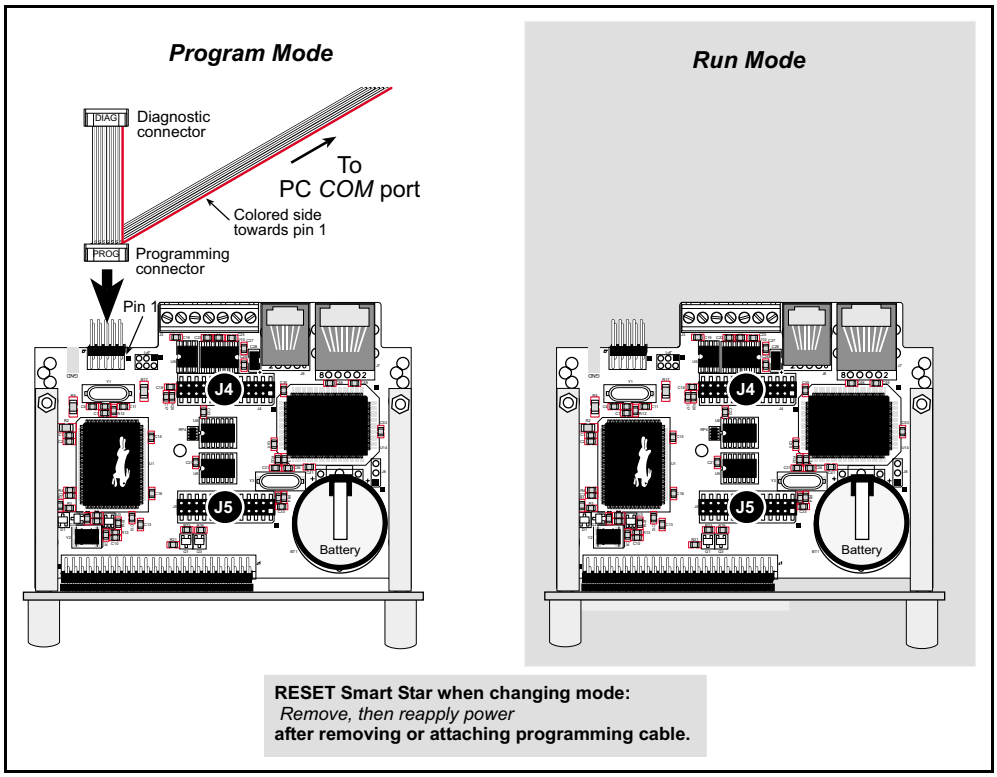


Figure 17. Smart Star Program Mode and Run Mode Set-Up

4.1.2 Changing from Program Mode to Run Mode

1. Disconnect the programming cable from header J2 of the CPU card.
2. Reset the Smart Star by unplugging the power supply, then plugging it back in.

The Smart Star is now ready to operate in the Run Mode.

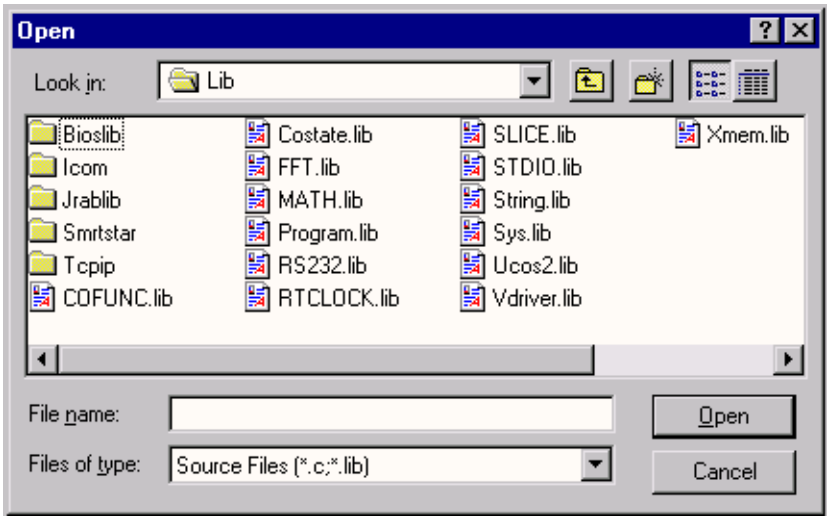
4.1.3 Changing from Run Mode to Program Mode

1. Attach the programming cable to header J2 of the CPU card.
2. Reset the Smart Star by unplugging the power supply, then plugging it back in.

The Smart Star is now ready to operate in the Program Mode.

4.2 Dynamic C Libraries

With Dynamic C running, click **File** > **Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.



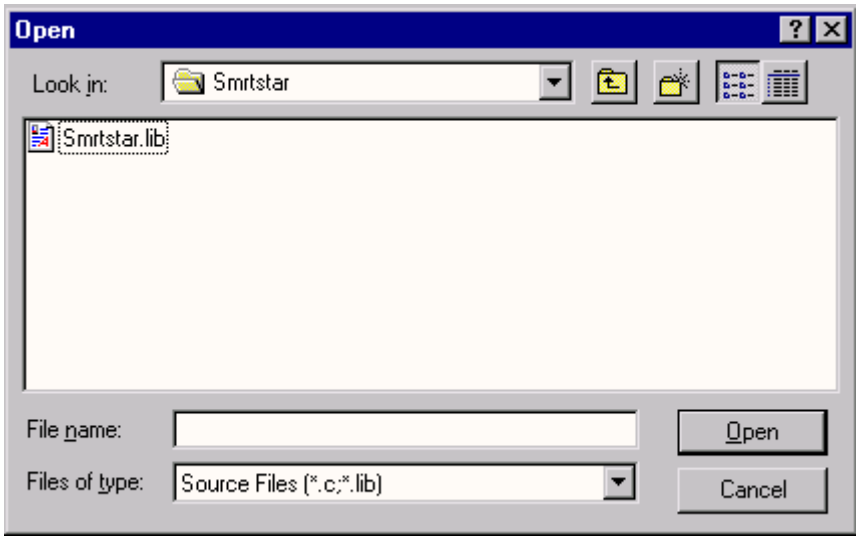
One library directory contains software that is unique to the Smart Star.

- **SMRTSTAR**—libraries associated with features specific to the Smart Star control system.

Other functions applicable to all devices based on the Rabbit 2000 microprocessor are described in the ***Dynamic C Premier User's Manual***.

4.2.1 Library Directories

The **SMRTSTAR** directory contains libraries required to operate the Smart Star control system.



- **SMRTSTAR.LIB**—This library supports all the functions needed by the Smart Star systems including digital I/O cards, relay cards, D/A converter and A/D converter cards, and serial communication.

Functions dealing with the backplane and the CPU card are described in this chapter. Functions relevant to the individual I/O cards are described in the chapter specific to the I/O card.

4.3 Smart Star Backplane Function APIs

4.3.1 Board Reset

```
void brdResetBus();
```

Resets all cards on the bus.

RETURN VALUE

None.

4.3.2 Board Initialization

```
void brdInit();
```

Initializes slot addressing, disables card enable/disable line, resets card slot bus and LED latch, and turns all LEDS OFF. Call this function at the beginning of the application.

RETURN VALUE

None.

4.4 Serial Communication APIs

Library files included with Dynamic C provide a full range of serial communications support. The **RS232.LIB** library provides a set of circular-buffer-based serial functions. The **PACKET.LIB** library provides packet-based serial functions where packets can be delimited by the 9th bit, by transmission gaps, or with user-defined special characters. Both libraries provide blocking functions, which do not return until they are finished transmitting or receiving, and nonblocking functions, which must be called repeatedly until they are finished. For more information, see the ***Dynamic C Premier User's Manual*** and Technical Note 213, ***Rabbit 2000 Serial Port Software***.

Use the following function calls with the Smart Star.

int serMode(int mode);

User interface to set up up serial communication lines for the Smart Star control system. Call this function after **serXOpen()**.

PARAMETERS

mode is the defined serial port configuration of the CPU card.

Mode	Serial Port		Parallel Port
	C (PC2 and PC3)	D (PC0 and PC1)	D (PD0 and PD1)
0	RS-232, 3-wire	RS-485	
1	RS-232, 3-wire	RS-232, 3-wire	
2	RS-232, 5-wire		RTS/CTS
3	RS-232, 5-wire	RS-485	RTS/CTS

RETURN VALUE

0 if correct mode, 1 if not.

ser485Tx();

Enables RS-485 transmission (disables receive) on serial port D.

RETURN VALUE

None.

SEE ALSO

ser485Rx

ser485Rx();

Disables RS-485 transmission (enables receive) on serial port D.

RETURN VALUE

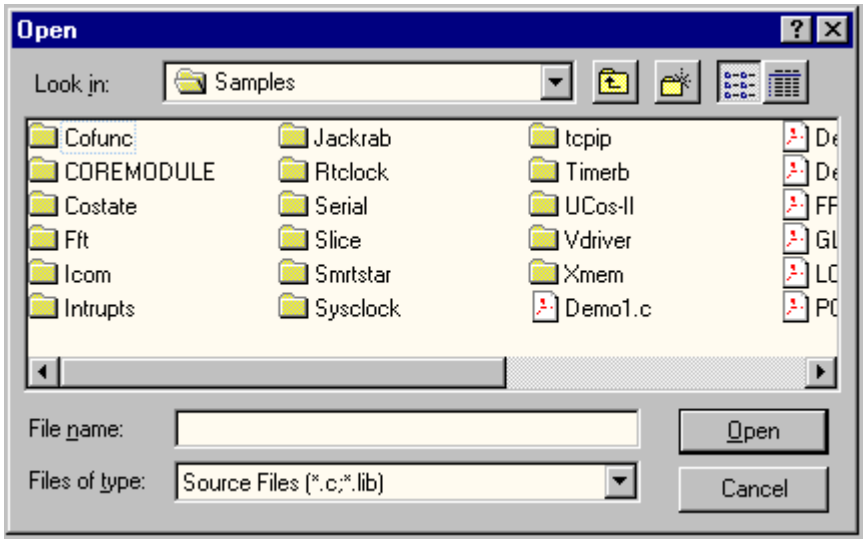
None.

SEE ALSO

ser485Tx

4.5 Sample Programs

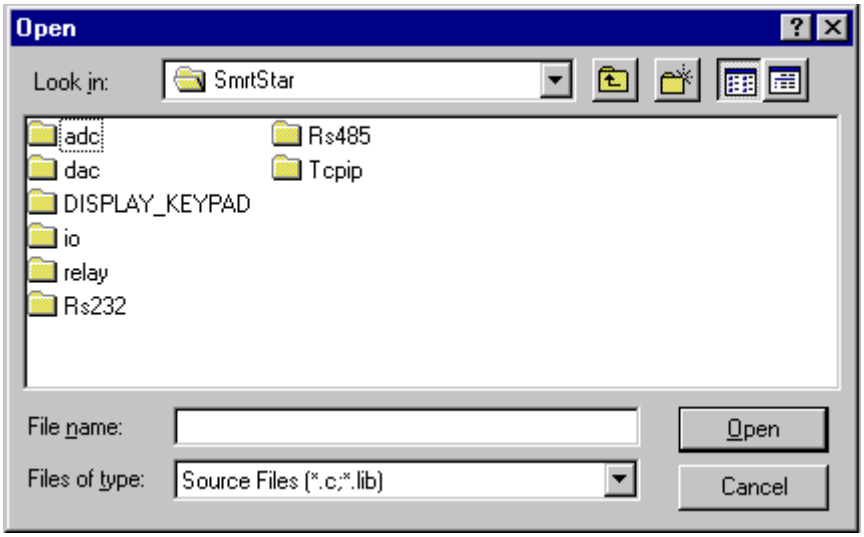
Sample programs are provided in the Dynamic C **Samples** folder, which is shown below.



The various folders contain specific sample programs that illustrate the use of the corresponding Dynamic C libraries. For example, the sample program **PONG.C** demonstrates the output to the **STDIO** window.

The **SMRTSTAR** folder provides sample programs specific to the Smart Star control system. Each sample program has comments that describe the purpose and function of the program. Follow the instructions at the beginning of the sample program.

Let's take a look at sample programs for the backplane and the CPU card in the **SMRT-STAR** folder.



The **RS232** directory contains two sample programs to illustrate RS-232 serial communication.

- **SSTAR232.C**—Demonstrates a simple RS-232 loopback using both serial ports C and D.
- **SSTAR5W.C**—Demonstrates simple 5-wire RS-232 communication with flow control.

The **RS485** directory contains two sample programs to illustrate RS-485 serial communication.

- **MASTER.C**—Demonstrates a simple RS-485 transmission of lower case letters to a slave controller. The slave will send converted upper case letters back to the master controller for display in the **STDIO** window. Use **SLAVE.C** to program the slave controller.
- **SLAVE.C**—Demonstrates a simple RS-485 transmission of alphabetic characters to a master controller. The slave will send converted upper case letters back to the master controller for display in the **STDIO** window. Use **MASTER.C** to program the master controller.

4.6 Using Dynamic C

To run a sample program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The CPU card must be in Program Mode (see Section 4.1, "Programming Cable") and must be connected to a PC using the programming cable as described in Section 2.3, "Programming Cable Connections."

More complete information on Dynamic C is provided in the ***Dynamic C Premier User's Manual***.

5. USING THE TCP/IP FEATURES

Chapter 5 discusses using the TCP/IP features on the CPU cards. Note that the TCP/IP feature is available only on the SR9150 CPU card.

5.1 Ethernet Connections

Before proceeding you will need to have the following items.

- If you don't have an Ethernet connection, you will need to install a 10Base-T Ethernet card (available from your favorite computer supplier) in your PC.
- Two RJ-45 straight-through Ethernet cables and a hub, or an RJ-45 crossover Ethernet cable.

The Ethernet cables and Ethernet hub are available from Z-World in a TCP/IP tool kit. More information is available at www.zworld.com.

1. **Install the CPU card on the backplane, and connect the power supply and the programming cable as shown in Chapter 2, "Getting Started."**
2. **Ethernet Connections**

- If you do not have access to an Ethernet network, use a crossover Ethernet cable to connect the installed CPU card to a PC that at least has a 10Base-T Ethernet card.
- If you have an Ethernet connection, use a straight-through Ethernet cable to establish an Ethernet connection to the installed CPU card from an Ethernet hub. These connections are shown in Figure 18.

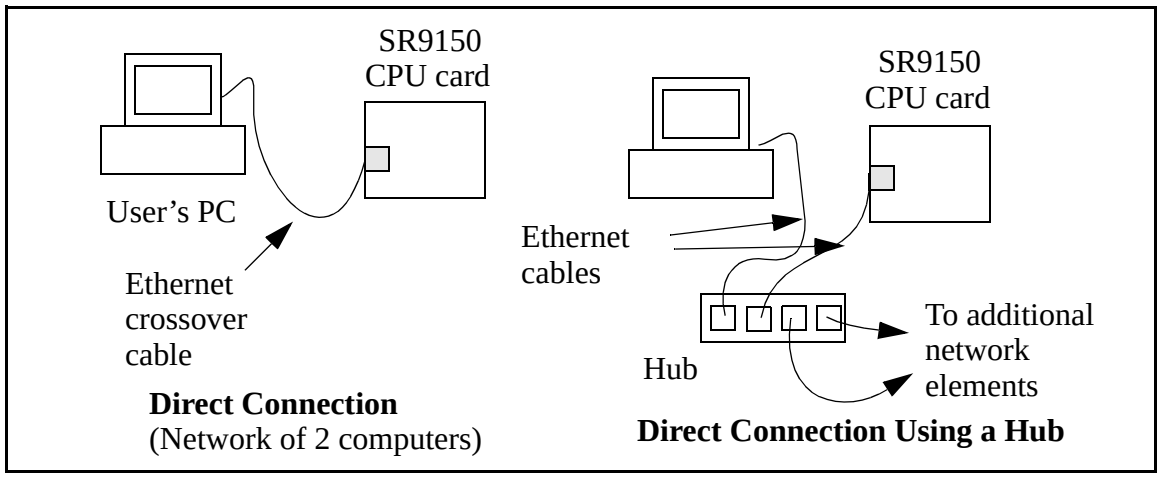


Figure 18. Ethernet Connections

3. Apply Power

Plug in the power supply. The Smart Star is now ready to be used.

> **NOTE:** A hardware RESET is accomplished by unplugging the power supply, then plugging it back in.

The green **LNK** light is on the CPU card is on when the Smart Star is properly connected either to an Ethernet hub or to an active Ethernet card. The orange **ACT** light flashes each time a packet is received.

5.2 TCP/IP Sample Programs

We have provided a number of sample programs demonstrating various uses of TCP/IP for networking embedded systems. These programs require that you connect your PC and the Smart Star together on the same network. This network can be a local private network (preferred for initial experimentation and debugging), or a connection via the Internet.

5.2.1 How to Set IP Addresses in the Sample Programs

Most of the sample programs use macros to define the IP address assigned to the CPU card and the IP address of the gateway, if there is a gateway.

```
#define MY_IP_ADDRESS "216.112.116.155"
#define MY_NETMASK "255.255.255.248"
#define MY_GATEWAY "216.112.116.153"
```

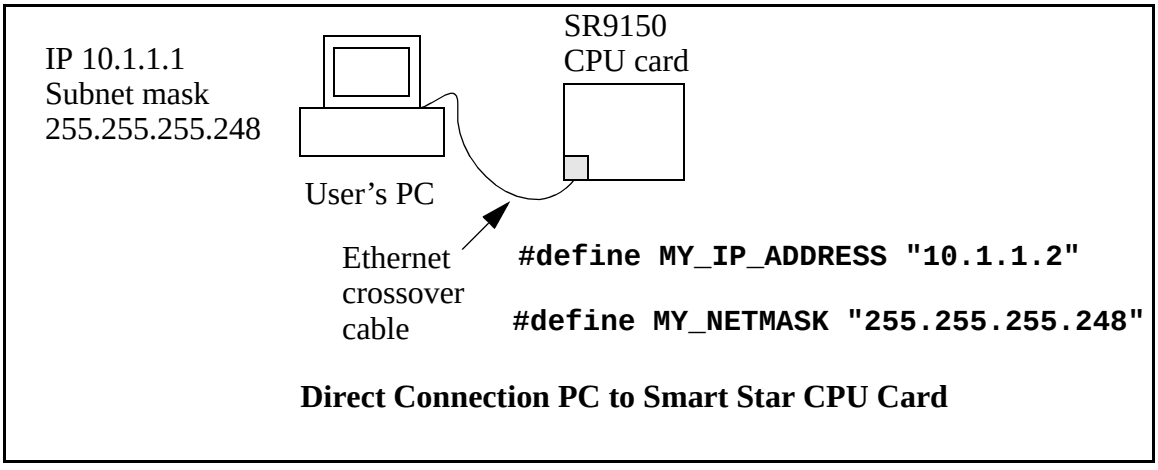
In order to do a direct connection, the following IP addresses can be used for the CPU card:

```
#define MY_IP_ADDRESS "10.1.1.2"
#define MY_NETMASK "255.255.255.248"
// #define MY_GATEWAY "216.112.116.153"
```

In this case, the gateway is not used and is commented out. The IP address of the board is defined to be 10.1.1.2. The IP address of your PC can be defined as 10.1.1.1.

5.2.2 How to Set Up your Computer's IP Address for a Direct Connection

When your computer is connected directly to the OP6800 via an Ethernet connection, you need to assign an IP address to your computer. To assign the PC the address 10.1.1.1 with the subnetmask 255.255.255.248 under Windows 98, do the following.



Direct Connection PC to Smart Star CPU Card

Click on **Start > Settings > Control Panel** to bring up the Control Panel, and then double-click the Network icon. In the window find the line of the form **TCP/IP > Ethernet adapter name**. Double-click on this line to bring up the TCP/IP properties dialog box. You can edit the IP address directly and the subnet mask. (Disable "obtain an IP address automatically.") You may want to write down the existing values in case you have to restore them later. It is not necessary to edit the gateway address since the gateway is not used with direct connect.

The method of setting the IP address may differ for different versions of Windows, such as 95, NT, 2000, or XP.

5.2.3 Run the PINGME.C Demo

In order to run this program, edit the IP address and netmask in the **PINGME.C** program (**SAMPLES\TCPIP\ICMP**) to the values given above (10.1.1.2 and 255.255.255.248). Compile the program and start it running under Dynamic C. The crossover cable is connected from your computer's Ethernet adapter to the CPU card's RJ-45 Ethernet connector. When the program starts running, the green **LNK** light on the CPU card should be on to indicate that an Ethernet connection is made. (Note: If the **LNK** light does not light, you may not have a crossover cable, or if you are using a hub perhaps the power is off on the hub.)

The next step is to ping the board from your PC. This can be done by bringing up the MS-DOS window and running the ping program:

```
ping 10.1.1.2
```

or by **Start > Run**

and typing the command

```
ping 10.1.1.2
```

Notice that the orange **ACT** light flashes on the CPU card while the ping is taking place, and indicates the transfer of data. The ping routine will ping the board four times and write a summary message on the screen describing the operation.

5.2.4 Additional Demo Programs

The program **SMTP.C** (**SAMPLES\SMRTSTAR\TCPIP**) demonstrates a basic Smart Star system using the SMTP library to send an e-mail when a keypress is detected on an LCD/keypad module. In order to run this sample program, edit the IP address as for the pingme program, edit the "mail to" e-mail address, compile the program, and start it executing. An e-mail corresponding to the keypad button that was pressed is sent.

The program **SSI.C** (**SAMPLES\SMRTSTAR\TCPIP**) demonstrates how to make the Smart Star CPU card a Web server. This program allows you to turn the LEDs on an attached LCD/keypad module on and off from a remote Web browser. In order to run these sample programs, edit the IP address as for the pingme program, compile the program, and start it executing. Then bring up your Web browser and enter the following server address: http://10.1.1.2. This should bring up the Web page served by the sample program.

The program **SSI2.C** (**SAMPLES\SMRTSTAR\TCPIP**) demonstrates the use of I/O cards via instructions sent from a Web browser. You will need an A/D converter I/O card, a D/A converter I/O card, or a relay I/O card installed on the backplane in order for the Web browser to be able to initiate changes on one or more of these I/O cards. Before you run this sample program, edit the IP address as for the pingme program, compile the program, and start it executing. The analog outputs will change or the relays will open and close in response to instructions sent from the Web browser.

5.2.5 LCD/Keypad Sample Programs Showing TCP/IP Features

The following sample programs, found in the **TCPIP** subdirectory in **SAMPLES/LCD_Keypad/122x32_1x7**, are demonstrate the features of the LCD/keypad module connected to the backplane. Remember to configure the IP address, netmask, and gateway as indicated in the sample programs.

- **MBOXDEMO.C**—This program implements a web server that allows Web e-mail messages to be entered that are then shown on the LCD display. The keypad allows you to scroll within messages, flip to other e-mails, mark messages as read, and delete e-mails. When a new e-mail arrives, an LED turns on, and turns off once the message has been marked as read. A log of all e-mail actions is kept, and can be displayed in the Web browser. All current e-mails can also be read with the Web browser.

  When using **MBOXDEMO.C**, connect the Smart Star CPU card and a PC (or other device with a Web Browser) to an Ethernet. If you connect the PC and the CPU card directly, be sure to use a crossover Ethernet cable; straight-through Ethernet cables and a hub may be used instead.

- **TCP_RESPOND.C**—This program and **TCP_SEND.C** are executed on two separate single-board computers to demonstrate how the two boards communicate with each other. Use **PCSEND.EXE** on the PC console side at the command prompt if you do not have a second board. **PCSEND.EXE** is located with source code in the **SAMPLES/LCD_Keypad/Windows** directory.

  TCP_RESPOND.C waits for a message from another single-board computer. The message received is displayed on the LCD, and you may respond by pressing a key on the keypad. The response is then sent to the remote single-board computer.

- **TCPSEND.C**—This program and **TCP_RESPOND.C** are executed on two separate single-board computers to demonstrate how the two boards communicate with each other. Use **PCRESPOND.EXE** on the PC console side at the command prompt if you do not have a second board. **PCRESPOND.EXE** is located with source code in the **SAMPLES/LCD_Keypad/Windows** directory.

  When a key on the keypad is pressed, a message associated with that key is sent to a specified destination address and port. The destination then responds to that message. The response is displayed on the LCD.

  Note that only the **LEFT** and **UP** scroll keys are set up to cause a message to be sent.

When using **TCPSEND.C** and **TCP_RESPOND.C**, connect the CPU card and the other single-board computer to an Ethernet. If you connect the them directly, be sure to use a crossover Ethernet cable; straight-through Ethernet cables and a hub may be used instead.

5.3 Where Do I Go From Here?

NOTE: If you purchased your Smart Star through a distributor or Z-World partner, contact the distributor or Z-World partner first for technical support.

If there are any problems at this point:

- Check the Z-World Technical Bulletin Board at www.zworld.com/support/bb/.
- Use the Technical Support e-mail form at www.zworld.com/support/support_submit.html.
- Call Z-World Technical Support at (530)757-3737.

If the sample programs ran fine, you are now ready to go on.

Additional sample programs are described in the ***Dynamic C TCP/IP User's Manual***.

Refer to the ***Dynamic C TCP/IP User's Manual*** to develop your own applications. ***An Introduction to TCP/IP*** provides background information on TCP/IP, and is available on Z-World's Web site.

6. SMART STAR SPECIFICATIONS

This chapter provides the specifications for the Smart Star backplane and CPU card, and describes the conformal coating.

6.1 Electrical and Mechanical Specifications

6.1.1 Smart Star Backplane

Figure 19 shows the mechanical dimensions for the two Smart Star backplanes.

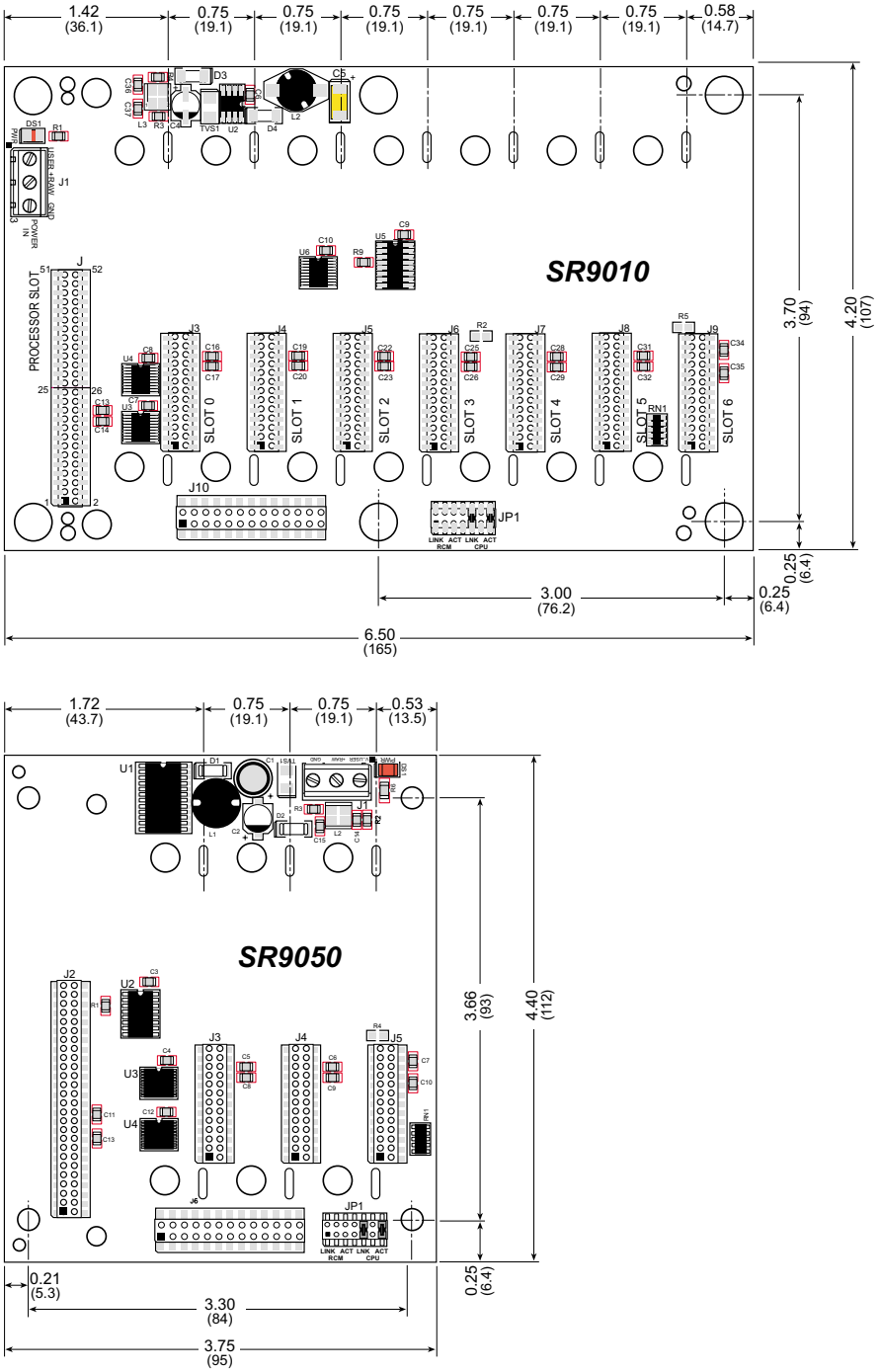


Figure 19. Smart Star Backplane Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

Table 3 lists the electrical, mechanical, and environmental specifications for the Smart Star backplanes.

Table 3. Smart Star Backplane Specifications

Parameter	Specification	
	SR9010	**SR9050**
Board Size	6.50" × 4.20" × 0.75" (165 mm × 107 mm × 19 mm)	3.75" × 4.40" × 0.75" (95 mm × 112 mm × 19 mm)
Connectors	one 2 × 26 (CPU card slot), 2 mm seven 2 × 13 (I/O card slots), 2 mm	one 2 × 26 (CPU card slot), 2 mm three 2 × 13 (I/O card slots), 2 mm
Slot Select	Each slot has a predefined dedicated set of addresses (see Appendix E and the software chapters in the individual I/O card manuals)	
Temperature	–40°C to +70°C	
Humidity	5% to 95%, noncondensing	
External Input Voltage	9 V to 30 V DC at 1 A typical for onboard +5 V regulated supply; provision for independent 9 V to 30 V DC (**V_USER**) voltage source for I/O cards—the exact voltage for the second supply depends on the requirements of the specific I/O cards used (Z-World recommends tying **V_USER** to **+RAW** unless there is a specific need for an independent power supply)	
Onboard Voltage Regulator	Surface-mount switching regulator sources 5 V at 1 A	
Data Lines	Buffered bidirectional data lines (D0–D7)	
Address Lines	Buffered address lines (A0–A3)	
Read/Write Control	Buffered IORD, IOWR	
Reset	I/O cards and CPU card can be reset independently	

6.1.2 CPU Card

Figure 20 shows the mechanical dimensions for the CPU cards.

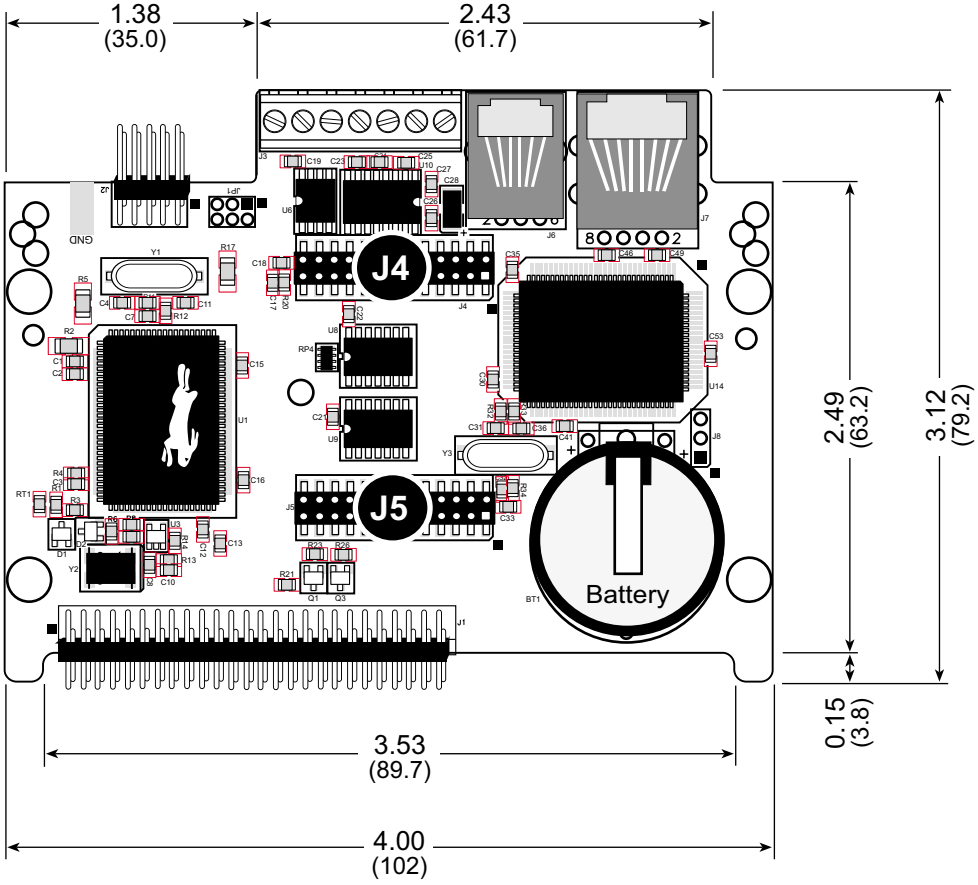


Figure 20. CPU Card Dimensions

Table 4 lists the electrical, mechanical, and environmental specifications for the CPU card.

Table 4. CPU Card Specifications

Parameter	Specification
Board Size (with optional backup battery board)	4.00" × 3.12" × 1.00" (102 mm × 79.2 mm × 25.4 mm)
Connectors	one RJ-45 (Ethernet) (SR9150 only) one 2 × 5, 2 mm pitch (serial programming port) one 0.9 mm × 0.5 screw-terminal connector strips (accept 14–30 AWG or 0.05–1.5 mm² wire)
Ethernet Interface (SR9150 only)	Direct connection to 10Base-T Ethernet networks via RJ-45 connection
Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Input Voltage	5 V DC at 190 mA typical
Microprocessor	Rabbit 2000™
Clock	22.1 MHz
SRAM	128K, surface mounted, 512K option
Flash EPROM	2 × 256K, surface mounted
Timers	Five 8-bit timers cascadable in pairs, one 10-bit timer with 2 match registers that each have an interrupt
Serial Ports	Three serial ports: • one CMOS-compatible programming port • remaining ports software-configurable as two 3-wire RS-232, one 5-wire RS-232, or one 3-wire RS-232/ one RS-485
Serial Rate	Selected baud rates up to 115, 200 bps CMOS-compatible port supports up to 6.45 Mbps (synchronous)
Watchdog/Supervisor	Yes
Time/Date Clock	Yes
Expansion Port	Supports up to 7 I/O cards
Backup Battery	Yes: Panasonic CR2330 or equivalent 3 V lithium coin type, 265 mA·h standard using onboard battery holder; provision for external battery

6.2 Jumper Configurations

Figure 21 shows the header locations used to configure the various CPU card options via jumpers.

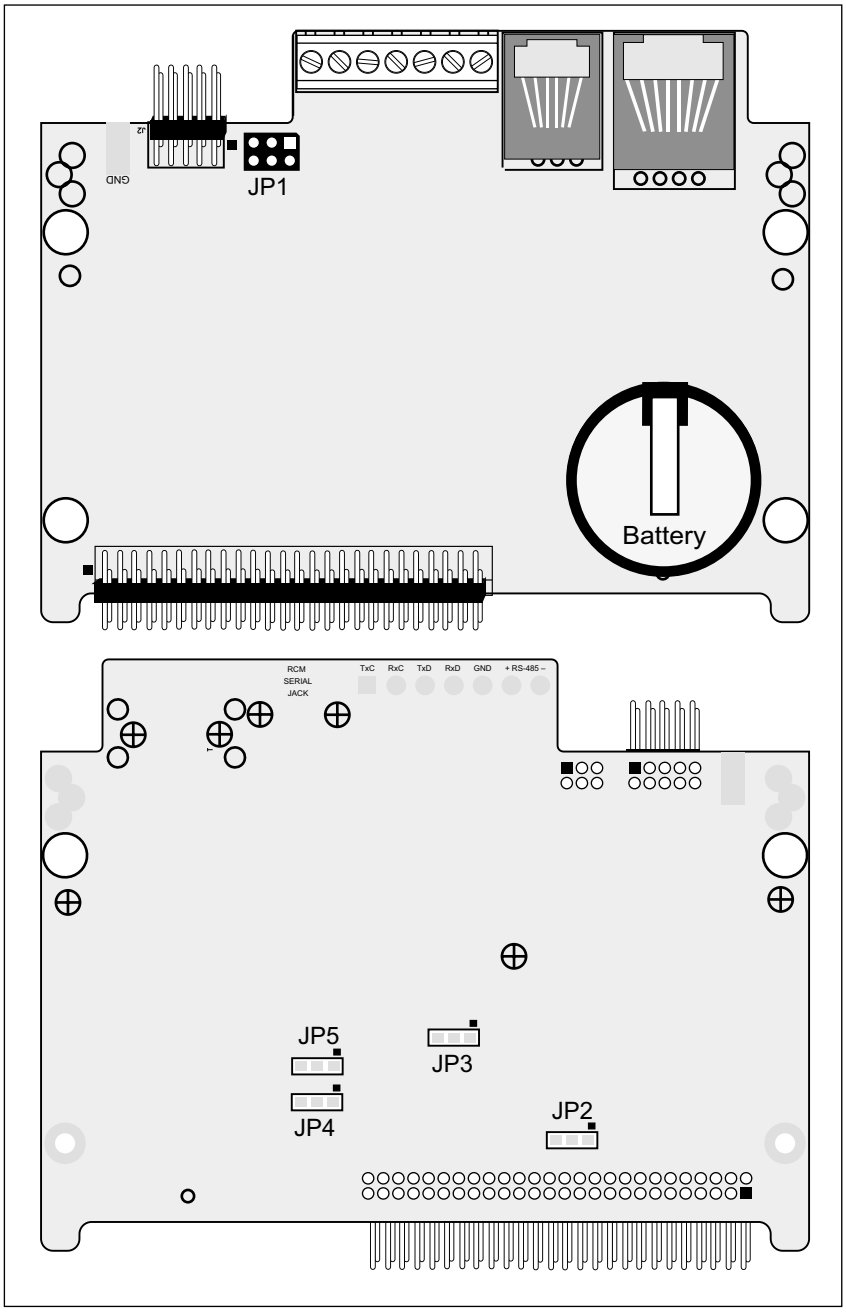


Figure 21. Location of Smart Star CPU Card Configurable Positions

Table 5 lists the configuration options.

Table 5. Smart Star CPU Card Jumper Configurations

Header	Description	Pins Connected		Factory Default
JP1	RS-485 Bias and Termination Resistors	1–2 5–6	Bias and termination resistors connected	×
		1–3 4–6	Bias and termination resistors ***not*** connected*	
JP2	U5 Flash Memory Size	1–2	128K/256K	×
		2–3	512K	
JP3	SRAM Size	1–2	128K	×
		2–3	512K	
JP4	U11 Flash Memory Size	1–2	128K/256K	×
		2–3	512K	
JP5	Flash Memory Bank Select	1–2	Normal Mode	×
		2–3	Bank Mode	

* Although pins 1–3 and 4–6 of header JP1 are shown "jumpered" for the termination and bias resistors ***not*** connected, pins 3 and 4 are not actually connected to anything, and this configuration is a "parking" configuration for the jumpers so that they will be readily available should you need to enable the termination and bias resistors in the future.

6.3 Conformal Coating

The areas around the crystal oscillator and the battery backup circuit on the CPU card have had the Dow Corning silicone-based 1-2620 conformal coating applied. The conformally coated areas are shown in Figure 22. The conformal coating protects these high-impedance circuits from the effects of moisture and contaiminants over time.

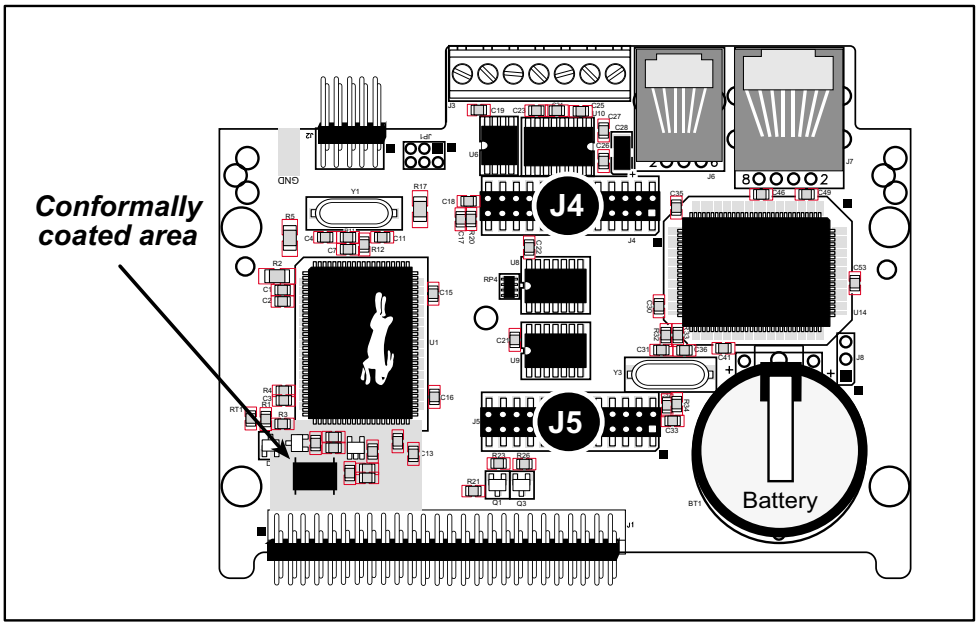


Figure 22. CPU Card Areas Receiving Conformal Coating

Any components in the conformally coated area may be replaced using standard soldering procedures for surface-mounted components. A new conformal coating should then be applied to offer continuing protection against the effects of moisture and contaminants.

NOTE: For more information on conformal coatings, refer to Rabbit Semiconductor Technical Note 303, ***Conformal Coatings***.

6.4 Use of Rabbit 2000 Parallel Ports

Figure 23 shows the Rabbit 2000 parallel ports.

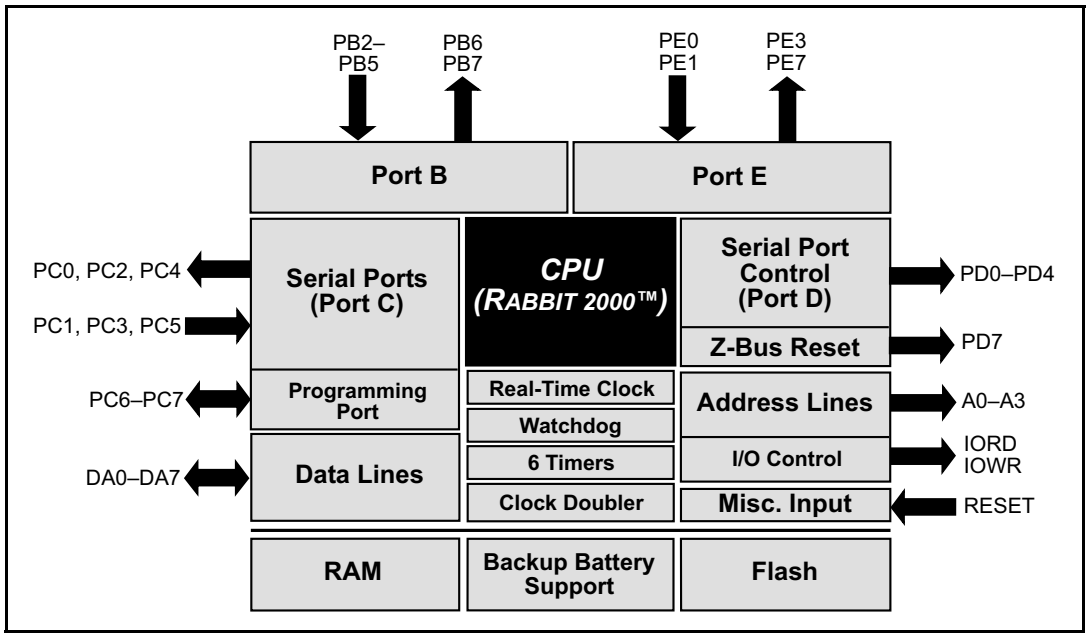


Figure 23. Smart Star CPU Card Rabbit 2000 Systems

6.5 Exclusion Zone

It is recommended that you allow for an “exclusion zone” of 3" (80 mm) around the Smart Star in all directions when the Smart Star is incorporated into an assembly that includes other components. This “exclusion zone” that you keep free of other components and boards will allow for sufficient air flow, and will help to minimize any electrical or EMI interference between adjacent boards.

PART II. DIGITAL I/O CARDS

7. DIGITAL I/O CARDS

Chapter 7 describes the features of the digital I/O card, one of the I/O cards designed for the Smart Star embedded control system.The Smart Star is a modular and expandable embedded control system whose configuration of I/O, A/D converter, D/A converter, and relay cards can be tailored to a large variety of demanding real-time control and data acquisition applications.

The typical Smart Star system consists of a rugged backplane with a power supply, a CPU card, and one or more I/O cards. The CPU card plugs into a designated slot on the backplane chassis, which has seven additional slots available for I/O cards to be used in any combination. A high-performance Rabbit 2000 microprocessor on the CPU card operates at 25.8 MHz to provide fast data processing.

7.1 Features

The SR9200 digital I/O cards offer protected digital inputs and high-current driver outputs in three banks, each containing 8 I/O points. One bank's configuration is fixed as protected digital inputs, one bank's configuration is fixed as high-current driver outputs, and one bank may be configured either as protected digital inputs or as high-current driver outputs, depending on the model of digital I/O card selected. The high-current driver outputs are either all sinking or all sourcing, depending on the model of digital I/O card selected.

Table 6 lists the digital I/O cards that are available for the Smart Star control system.

Table 6. Smart Star Digital I/O Cards

I/O Card	Model	Features
Digital I/O	SR9200	16 digital inputs, 8 digital sinking outputs
	SR9210	8 digital inputs, 16 digital sinking outputs
	SR9220	8 digital inputs, 8 digital sinking outputs
	SR9205	16 digital inputs, 8 digital sourcing outputs
	SR9215	8 digital inputs, 16 digital sourcing outputs
	SR9225	8 digital inputs, 8 digital sourcing outputs

7.2 User Interface

Figure 24 shows the complete pinout for the user interface on header J2. Note that pin 1 is indicated by a small arrow on the ribbon cable connector.

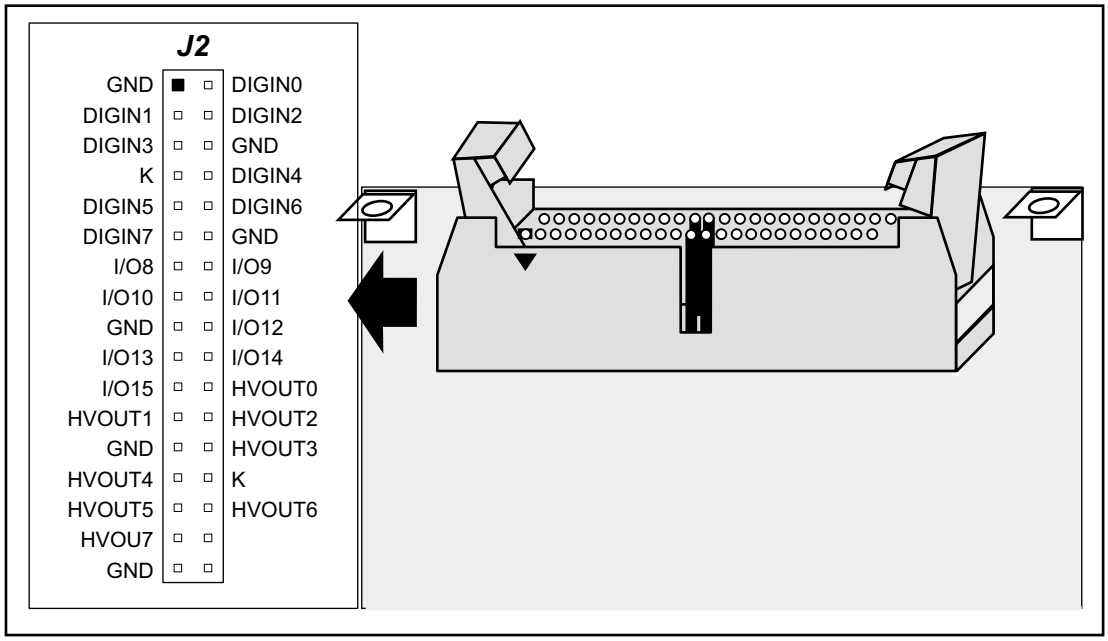


Figure 24. Digital I/O Card User Interface Pinout

7.3 User FWT Connections

Connections to the digital I/O cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Table 7 lists the Z-World part numbers for the FWTs.

Table 7. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals	Screw Terminals
			
FWT27	Digital I/O	101-0420	101-0424

7.3.1 Pinouts

Figure 25 shows the pinout for FWT27s used on digital I/O cards. Note that only 23 of the I/O points are available on the FWT27—the HVOUT7 digital output is not available on the FWT27.

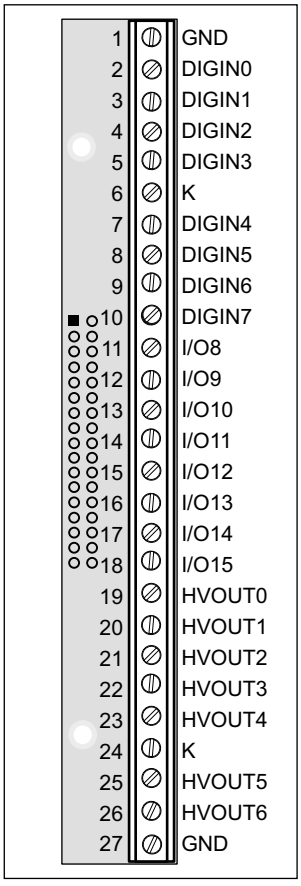


Figure 25. FWT Pinout for Digital I/O Cards

7.4 Digital Inputs and Outputs

The digital I/O card has 24 I/O points that are factory configured as either inputs or outputs in banks of eight, depending on the model.

Figure 26 shows the locations of the I/O banks.

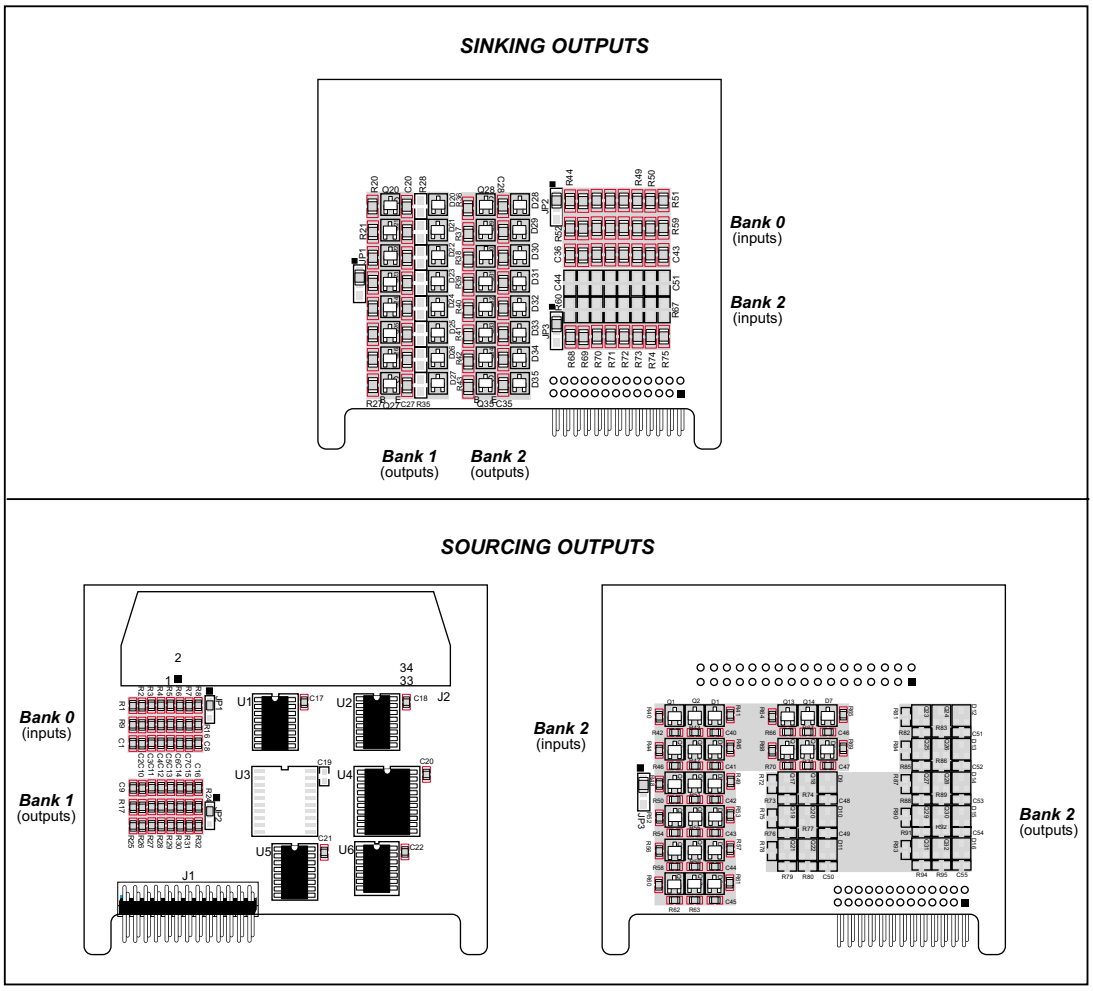


Figure 26. Locations of Banks

The I/O points on Bank 0 are always inputs, and the I/O points on Bank 1 are always outputs. The I/O points on Bank 2 were configured at the factory as either inputs *or* outputs, depending on the model of the digital I/O card. Table 8 lists the factory configurations.

Table 8. Digital I/O Card Bank 2 Factory Configurations

Model	Bank 2 Configured As
SR9200	Inputs
SR9210	Sinking outputs
SR9220	—
SR9205	Inputs
SR9215	Sourcing outputs
SR9225	—

The operation of Bank 2 is determined by the components on the digital I/O card. There is no jumper setting to select between inputs and outputs for Bank 2.

7.4.1 Digital Inputs

Table 9 provides the pinout configuration for the input points.

Table 9. Digital Inputs Pinout

Pin	Bank 0	Pin	Bank 2
2 DIGIN0	IN0	13 I/O8	IN8
3 DIGIN1	IN1	14 I/O9	IN9
4 DIGIN2	IN2	15 I/O10	IN10
5 DIGIN3	IN3	16 I/O11	IN11
8 DIGIN4	IN4	18 I/O12	IN12
9 DIGIN5	IN5	19 I/O13	IN13
10 DIGIN6	IN6	20 I/O14	IN14
11 DIGIN7	IN7	21 I/O15	IN15

The protected digital inputs, shown in Figure 27, are factory configured with 10 kΩ pull-up resistors. Digital I/O cards are also available in quantity with the protected digital inputs pulled down as shown in Figure 27.

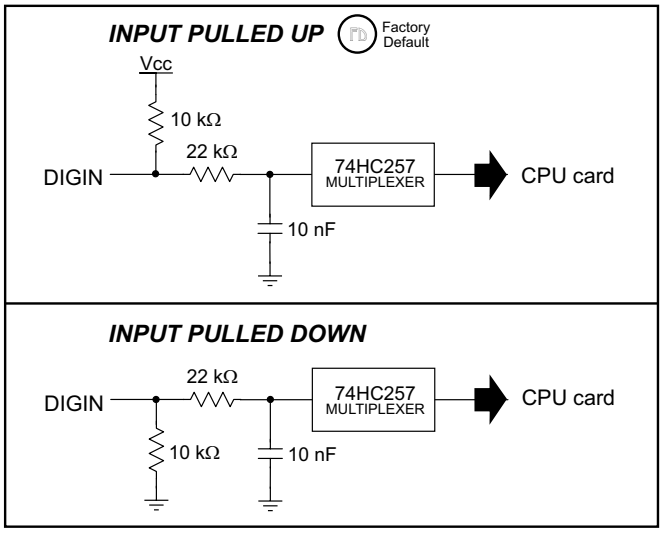


Figure 27. Protected Digital Inputs

A 0 Ω surface-mount resistor is used as a jumper to select whether the inputs are pulled up or down, as shown in Figure 28.

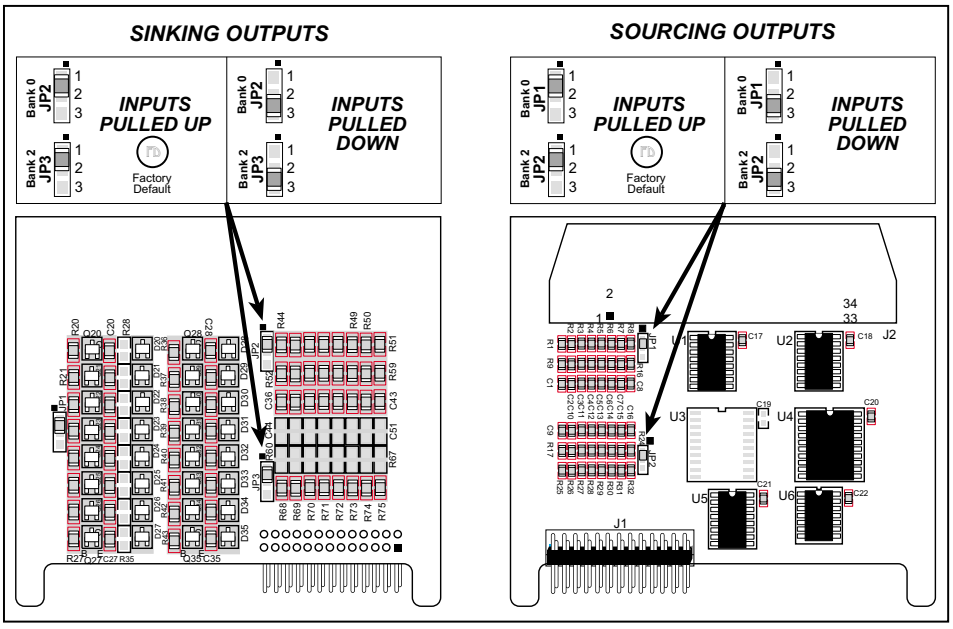


Figure 28. Selecting Pulled Up or Pulled Down Digital Inputs

The digital inputs are able to operate continuously from -30 V to +30 V, and have a logic threshold of 2.5 V. They are protected against spikes up to ±48 V.

7.4.2 Digital Outputs

The high-current digital outputs are either sinking or sourcing, depending on the model of the digital I/O card. Table 10 provides the pinout configuration for the output points.

Table 10. Digital Outputs Pinout

Pin	Bank 2	Pin	Bank 1
13 I/O8	OUT8	22 HVOUT0	OUT0
14 I/O9	OUT9	23 HVOUT1	OUT1
15 I/O10	OUT10	24 HVOUT2	OUT2
16 I/O11	OUT11	26 HVOUT3	OUT3
18 I/O12	OUT12	27 HVOUT4	OUT4
19 I/O13	OUT13	29 HVOUT5	OUT5
20 I/O14	OUT14	30 HVOUT6	OUT6
21 I/O15	OUT15	31 HVOUT7	OUT7

Figure 29 shows the power distribution on the digital I/O card.

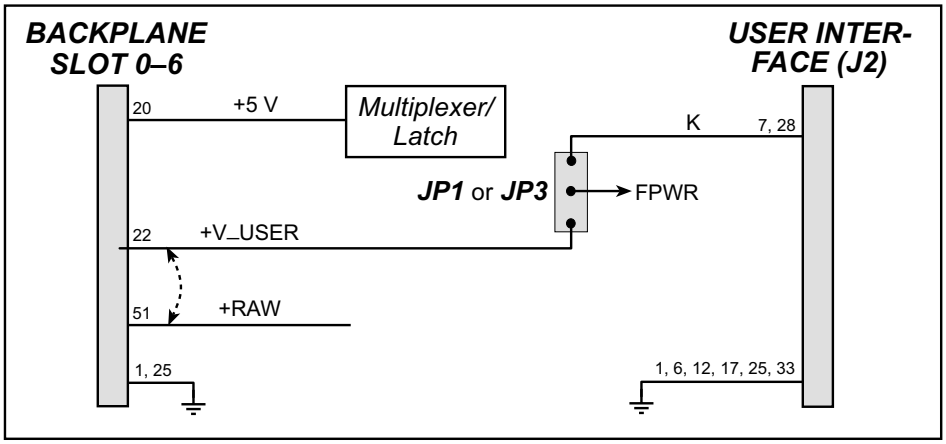


Figure 29. Digital I/O Card Power Distribution

When designing your interface with the Smart Star system, you need to establish whether you will use the **+V_USER/+RAW** supply on the backplane or your own independent **K** supply to drive the high-current outputs. The selection of this **FPWR** power supply is implemented via a 0 Ω surface-mount resistor on header JP1 (sinking outputs) or header JP3 (sourcing outputs) as shown in Figure 30. The factory default is to use **+V_USER/+RAW**, but digital I/O cards are available in quantity with the **FPWR** power supply jumpered to your own independent **K** supply.

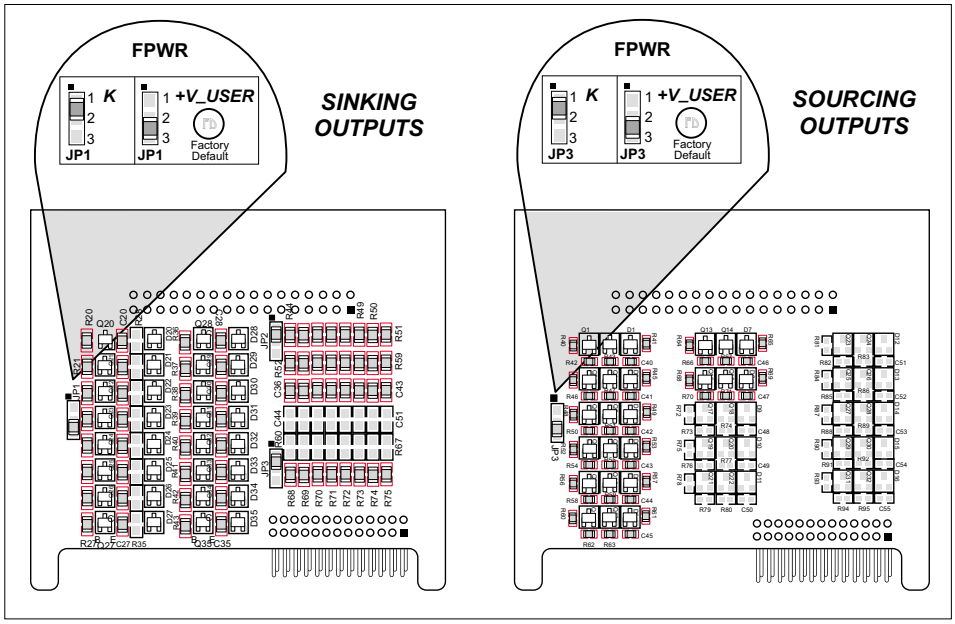


Figure 30. Selecting Power Supply for High-Current Sinking or Sourcing Outputs

Figure 31 shows how to connect a load to the high-current outputs based on whether your digital I/O card model has sinking or sourcing outputs.

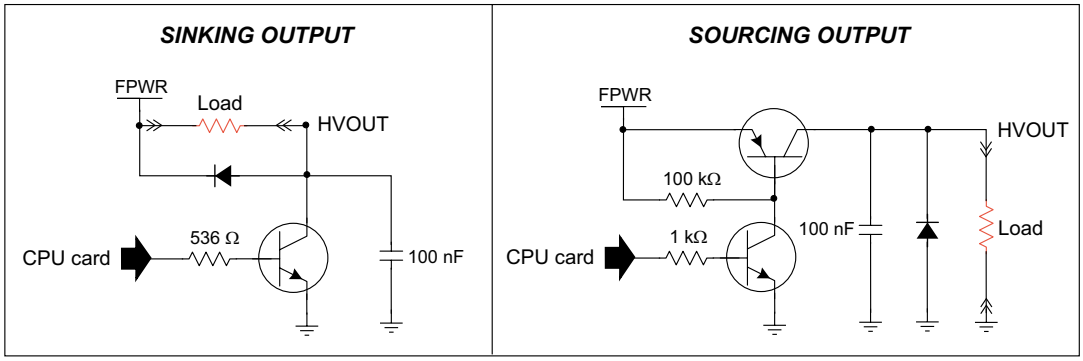


Figure 31. Connecting a Load to the High-Current Outputs

Each high-current output is able to sink or source up to 200 mA continuously, with a load limit of 40 V. Each high-current output may be switched independently, or a whole bank may be switched at once. The total current draw should be kept below 2.0 A when all high-current outputs on one digital I/O card are operating simultaneously, and the total current draw from your **+V_USER/+RAW** supply for all the I/O cards should be kept below 7.0 A.

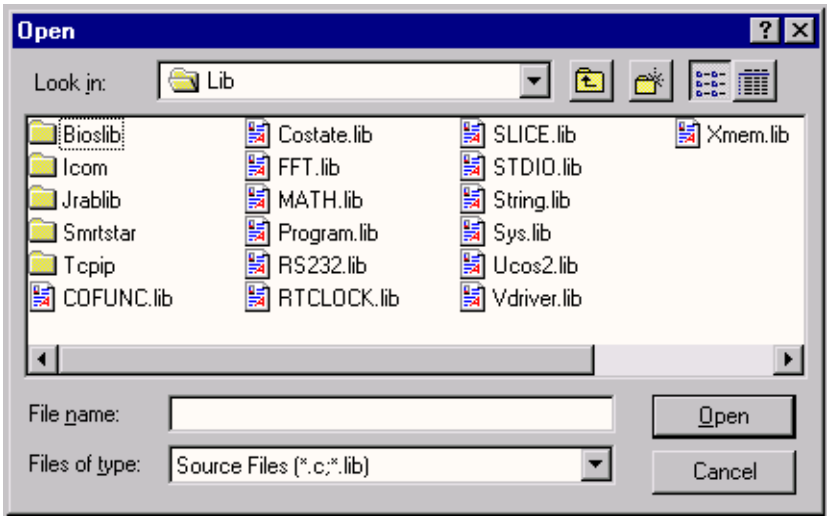
> **NOTE:** Note that the power supply provided in the Smart Star Tool Kit has a maximum output of 1.1 A.

7.5 Software

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

7.5.1 Dynamic C Libraries

With Dynamic C running, click **File > Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.



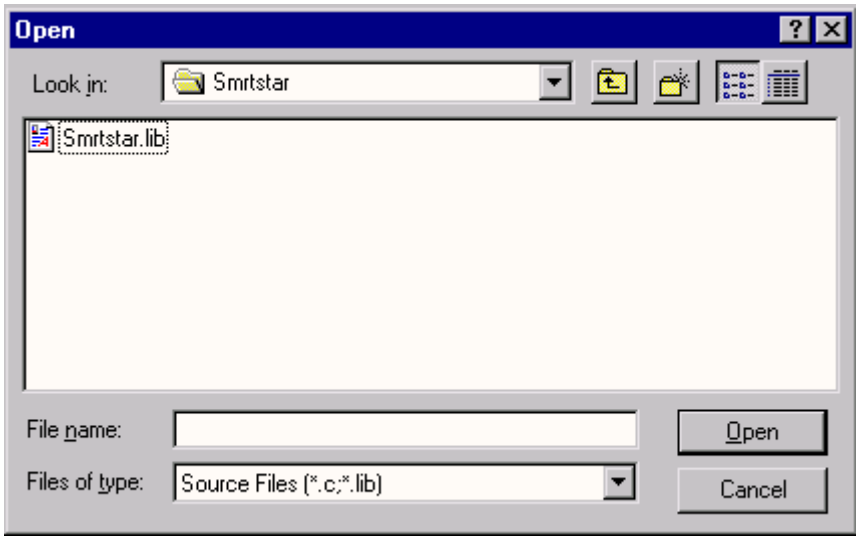
One library directory is specific to the Smart Star.

- **SMRTSTAR**—libraries associated with features specific to the Smart Star control system.

Other functions applicable to all devices based on the Rabbit 2000 microprocessor are described in the ***Dynamic C Premier User's Manual***.

7.5.2 Library Directories

The **SMRTSTAR** directory contains libraries required to operate the Smart Star control system.



- **SMRTSTAR.LIB**—This library supports all the functions needed by the Smart Star systems including digital I/O cards, relay cards, D/A converter and A/D converter cards, and serial communication.

7.5.3 Smart Star Digital I/O Card Function APIs

int digIn(int channel);

Reads the state of a digital input channel (IN0–IN15, IN8–IN15 is not available on all versions of the digital I/O card).

PARAMETER

channel is the digital input channel to read. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–15.

RETURN VALUE

The state of the digital input channel, 0 or 1.

SEE ALSO

digBankIn, **digOut**, **digBankOut**

int digBankIn(int bank);

Reads the state of Bank 0 or Bank 2 (if installed) digital input channels—Bank 0 consists of IN0–IN7 and Bank 2 consists of IN8–IN15.

PARAMETER

bank is the bank of digital input channels to read. **bank** should be passed as

```
bank = (slotnumber * 16) + (banknumber)
```

or

```
bank = BankAddr(slotnumber, banknumber)
```

where **slotnumber** is 0–6, and **banknumber** is 0 or 2.

RETURN VALUE

An input value in the lower byte, where each bit corresponds to one channel.

SEE ALSO

digIn, **digOut**, **digBankOut**

void digOut(int channel, int value);

Writes a value to an output channel (OUT0–OUT15, OUT8–IN15 not available on all versions of the digital I/O card).

PARAMETERS

channel is the digital output channel to write. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–15.

value is the output value, 0 or 1.

RETURN VALUE

None.

SEE ALSO

digBankOut, **digIn**, **digBankIn**

int digBankOut(int bank, int value);

Writes a byte value to Bank 1 or Bank 2 (if installed) digital output channels—Bank 1 consists of OUT0–OUT7 and Bank 2 consists of OUT8–OUT15.

PARAMETER

bank is the bank of digital output channels to write. **bank** should be passed as

```
bank = (slotnumber * 16) + (banknumber)
```

or

```
bank = BankAddr(slotnumber, banknumber)
```

where **slotnumber** is 0–6, and **banknumber** is 1 or 2.

value is the output value, where each bit correspons to one channel.

RETURN VALUE

An input value in the lower byte, where each bit corresponds to one channel.

SEE ALSO

digOut, digIn, digBankIn

7.5.4 Sample Programs

- **SSTARIO.C**—Demonstrates digital I/O using individual channels and whole banks. The sample program is set up for 8 inputs and 16 outputs. If necessary, you may change the macros in the sample program to match your digital I/O card.

7.5.4.1 Using Dynamic C

To run a sample program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The CPU card must be in Program Mode (see Section 4.1, "Programming Cable") and must be connected to a PC using the programming cable as described in Section 2.3, "Programming Cable Connections."

More complete information on Dynamic C is provided in the ***Dynamic C Premier User's Manual***.

7.6 Electrical and Mechanical Specifications

Figure 32 shows the mechanical dimensions for the digital I/O card.

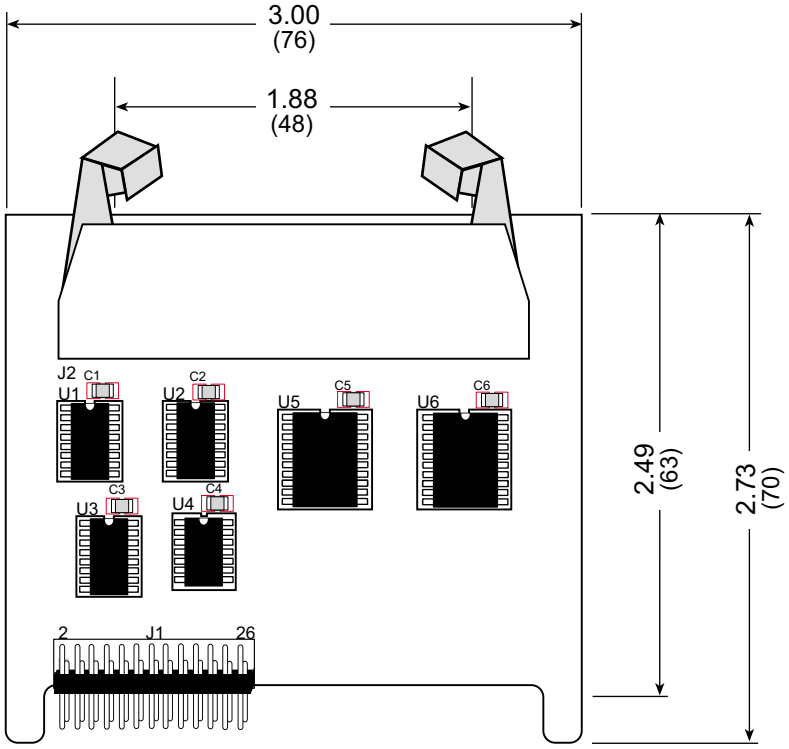


Figure 32. Digital I/O Card Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

Table 11 lists the electrical, mechanical, and environmental specifications for the digital I/O card.

Table 11. Digital I/O Card Specifications

Parameter	Specification
Board Size	2.73" × 3.00" × 0.44" (70 mm × 76 mm × 11 mm)
Connectors	one 2 × 17 latch/eject ribbon connector, 0.1 inch pitch
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Power Requirements	5 V DC at 65 mA from backplane (+5 V supply) 9 V to 30 V DC for **+RAW/+V_USER** from backplane or 9 V to 30 V DC for K on user interface header J2 Maximum draw 2.0 A from **+RAW/+V_USER** on backplane
Digital Inputs	Continuous operation from -30 V to +30 V, logic threshold at 2.5 V, protected against spikes ±48 V, 10 kΩ pull-up/pull-down resistors
Digital Outputs	Each output can sink (source) up to 200 mA continuously with load limit of 40 V, each output may be switched independently or bank of eight may be switched all at once, load current supplied from **+RAW/+V_USER** on backplane or user-supplied **K** on user interface header J2

PART III. A/D CONVERTER CARDS

8. A/D CONVERTER CARDS

Chapter 8 describes the features of the A/D converter card, one of the I/O cards designed for the Smart Star embedded control system.

The Smart Star is a modular and expandable embedded control system whose configuration of I/O, A/D converter, D/A converter, and relay cards can be tailored to a large variety of demanding real-time control and data acquisition applications.

The typical Smart Star system consists of a rugged backplane with a power supply, a CPU card, and one or more I/O cards. The CPU card plugs into a designated slot on the backplane chassis, which has seven additional slots available for I/O cards to be used in any combination. A high-performance Rabbit 2000 microprocessor on the CPU card operates at 25.8 MHz to provide fast data processing.

8.1 A/D Converter Card Features

Three models of A/D converter cards are available, as shown in Table 12.

Table 12. Smart Star A/D Converter Cards

I/O Card	Model	Features
A/D Converter	SR9300	12-bit A/D converter, 11 channels, 0 V – 10 V
	SR9310	12-bit A/D converter, 11 channels, -10 V – +10 V
	SR9320	12-bit A/D converter, 11 channels, 4 mA – 20 mA

8.2 User Interface

Figure 33 shows the circuit used to condition the analog signal before it goes to the A/D converter chip. Depending on the model of A/D converter card you have, it is designed to handle analog inputs of 0 V to 10 V, -10 V to +10 V, or 4–20 mA. The two different voltage ranges are handled with different gain resistors, R_g: 23.7 kΩ for the SR9300 and 12.1 kΩ for the SR9310. The input shown in Figure 33 is configured differently for the SR9320, which handles analog inputs of 4–20 mA.

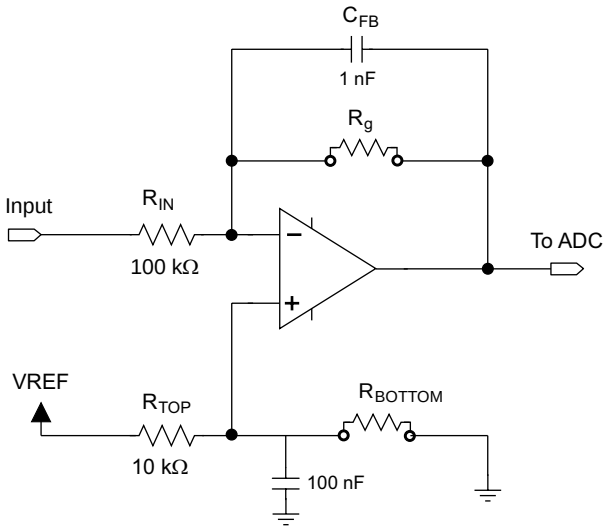


Figure 33. Analog Input Amplifier Circuit

Figure 34 shows the complete pinout for the user interface on header J2. Note that pin 1 is indicated by a small arrow on the ribbon cable connector.

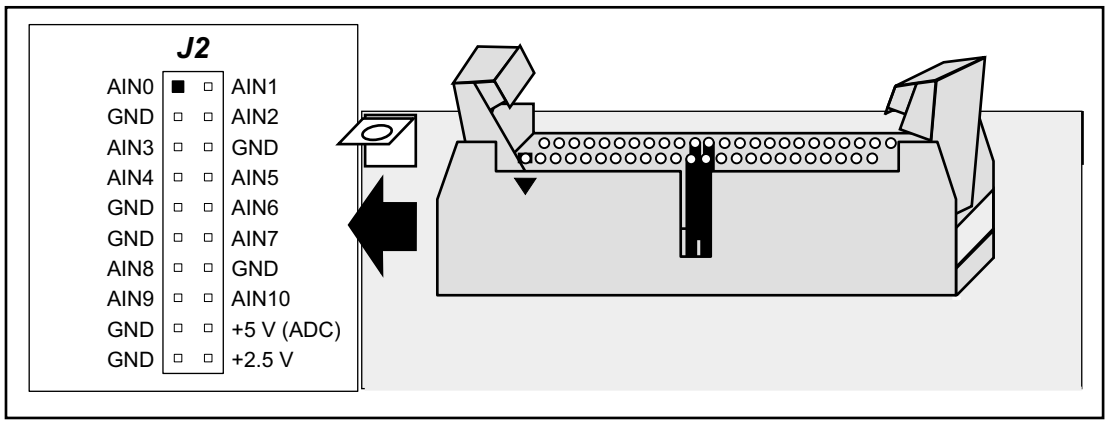


Figure 34. A/D Converter Card User Interface Pinout

8.3 User FWT Connections

Connections to the A/D converter cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Table 13 lists the the Z-World part numbers for the FWTs.

Table 13. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals	Screw Terminals
			
FWT18	A/D Converter	101-0421	101-0425

8.4 Pinouts

Figure 35 shows the pinout for the FWTs used on the A/D converter cards.

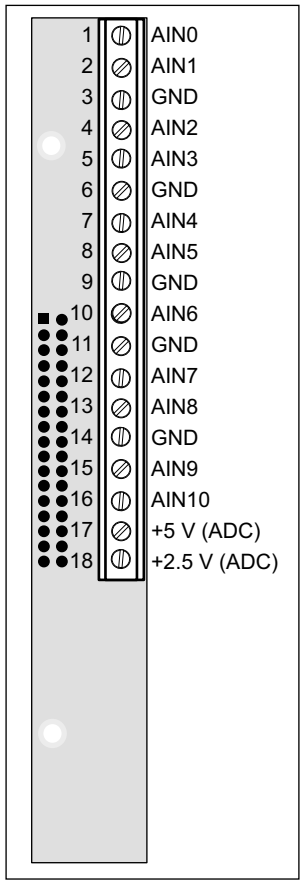


Figure 35. FWT Pinout for A/D Converter Cards

8.5 Power Distribution

Figure 36 shows the power distribution on the A/D converter card.

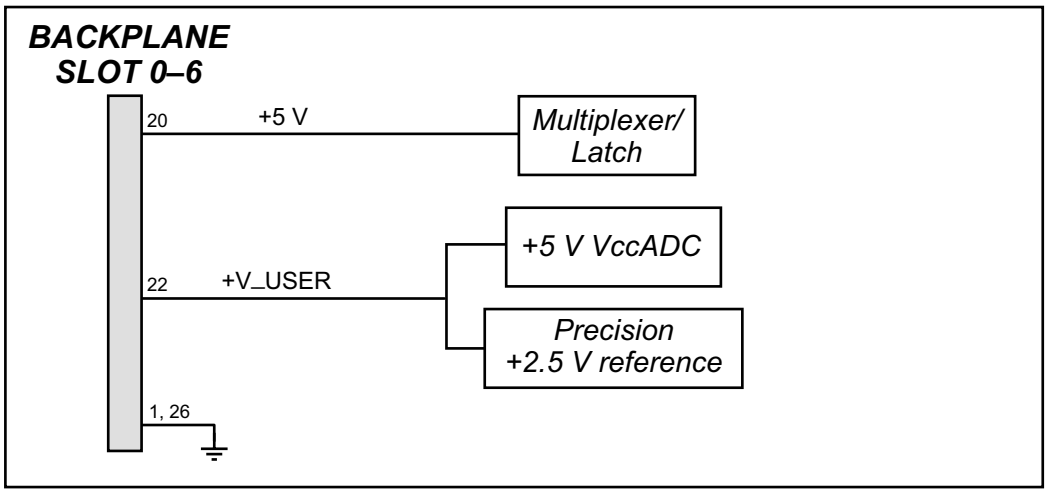


Figure 36. A/D Converter Card Power Distribution

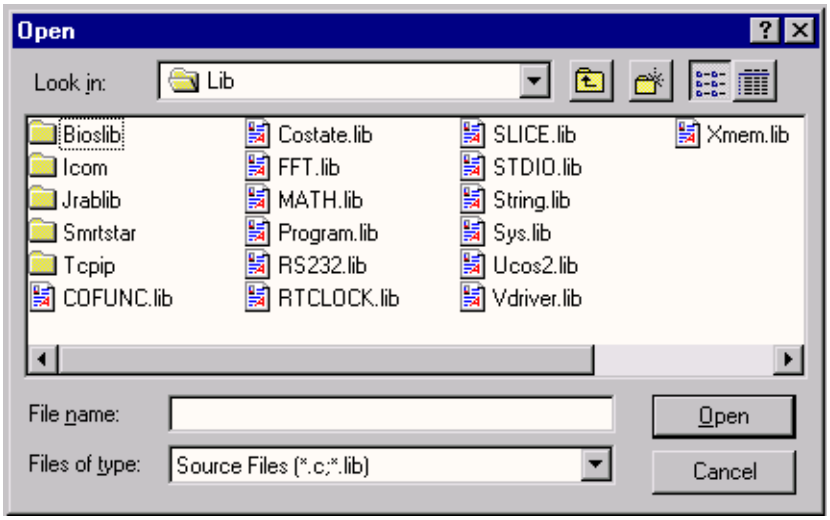
8.6 Software

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

8.6.1 Dynamic C Libraries

With Dynamic C running, click **File > Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.



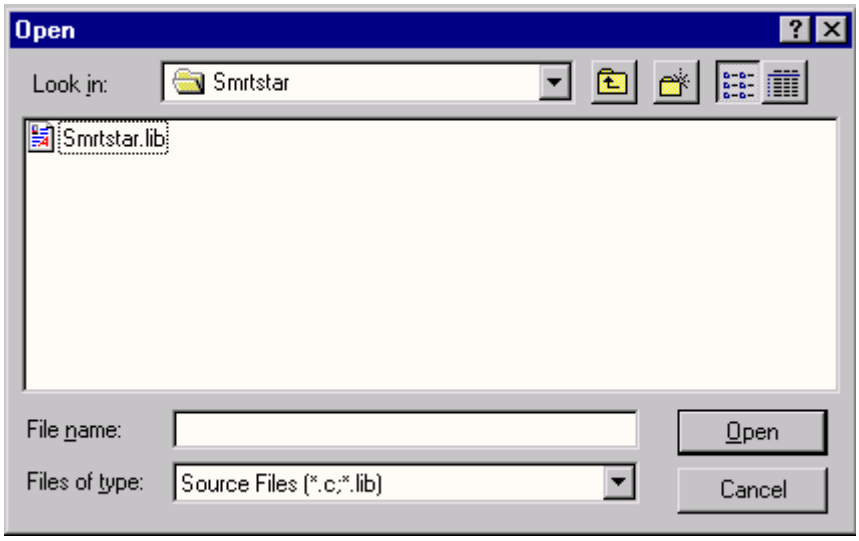
One library directory is specific to the Smart Star.

- **SMRTSTAR**—libraries associated with features specific to the Smart Star control system.

Other functions applicable to all devices based on the Rabbit 2000 microprocessor are described in the ***Dynamic C Premier User's Manual***.

8.6.2 Library Directories

The **SMRTSTAR** directory contains libraries required to operate the Smart Star control system.



- **SMRTSTAR.LIB**—This library supports all the functions needed by the Smart Star systems including digital I/O cards, relay cards, D/A converter and A/D converter cards, and serial communication.

8.6.3 Smart Star A/D Converter Card Function APIs

int anaInEERd(int channel);

The A/D converter card calibration constants, gain, and offset are stored in the factory in the upper half of the EEPROM on the A/D converter card. Use this function to read the A/D converter card calibration constants, gain, and offset from the upper half of the EEPROM on the A/D converter card

PARAMETERS

channel is the analog input channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10.

RETURN VALUE

0 if successful.
-1—control command unacceptable.
-2—EEPROM address unacceptable.

SEE ALSO

anaInEEWr

int anaSaveCalib();

The calibration constants may also be saved in the flash memory on the Smart Star CPU card. Doing so will speed up A/D conversions since a memory access from flash memory will be faster than from EEPROM. Use **anaSaveCalib** to save the current set of calibration constants for the analog input and output channels in the Smart Star flash memory. The calibration constants stored in flash memory can then be accessed at any time with the **anaLoadCalib** function.

If the factory-set calibration are not used, customer-measured calibration constants should first be established using the **anaInCalib** function.

RETURN VALUE

None.

SEE ALSO

anaLoadCalib, anaInCalib

void anaLoadCalib();

Reads a complete set of calibration constants for the analog input and output channels from the Smart Star flash memory on the CPU card. These should be set using the **anaInCalib** or **anaInEERd** function, then saved to flash memory using the **anaSaveCalib** function.

RETURN VALUE

None.

SEE ALSO

anaSaveCalib, anaInCalib

int anaInCalib(int channel, int value1, float volt1,int value2, float volt2);

Used to recalibrate the response of the A/D converter channel as a linear function using the two conversion points provided. Gain and offset constants are calculated and placed into the global table **_adcCalib**.

PARAMETERS

channel is the A/D converter input channel (0–10). **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10.

value1 is the first A/D converter value.

volt1 is the voltage corresponding to the first A/D converter value.

value2 is the second A/D converter value.

volt2 is the voltage corresponding to the first A/D converter value.

RETURN VALUE

0 if successful, -1, if not able to make calibration constants.

SEE ALSO

anaIn, anaInVolts

int anaInEEWr(int channel);

Writes the calibration constants, gain, and offset to the upper half of the EEPROM on the A/D converter card.

PARAMETERS

channel is the analog input channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10.

RETURN VALUE

0 if successful.
-1—control command unacceptable.
-2—EEPROM address unacceptable.
-3—data value unacceptable.

SEE ALSO

anaInEERd, _anaInEEWr

unsigned int anaIn(unsigned int channel);

Reads the state of an analog input channel and converts it to a digital value. A timeout occurs, causing the function to exit, if the end of the conversion is not detected within 13 µs.

PARAMETERS

channel is the analog input channel to read. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10.

RETURN VALUE

A value corresponding to the voltage on the analog input channel, 0–4095. A value outside this range indicates a failure

SEE ALSO

anaInCalib, anaInVolts

int anaInVolts(int channel);

Reads the state of an analog input channel and uses the previously set calibration constants to convert the state to volts.

PARAMETERS

channel is the analog input channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–10.

RETURN VALUE

A voltage value corresponding to the voltage on the analog input channel, 0–4095.

SEE ALSO

anaIn, anaInCalib

8.6.4 Sample Programs

- **SSTARAD1.C**—Demonstrates how to calibrate an A/D converter channel using two known voltages, and defines the two coefficients, gain and offset. These coefficients are then read back to compute the equivalent voltage.
- **SSTARAD2.C**—Reads and displays voltage and equivalent values of each A/D converter channel. Calibrations must have been previously stored into flash memory before running this program. See sample program **SSTARAD3.C**.
- **SSTARAD3.C**—Demonstrates how to calibrate all A/D converter channels using two known voltages and defines the two coefficients, gain and offset. These coefficients are then read back to compute the equivalent voltage and are saved to flash memory.

8.6.4.1 Using Dynamic C

To run a sample program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The CPU card must be in Program Mode (see Section 4.1, "Programming Cable") and must be connected to a PC using the programming cable as described in Section 2.3, "Programming Cable Connections."

More complete information on Dynamic C is provided in the ***Dynamic C Premier User's Manual***.

8.7 Electrical and Mechanical Specifications

Figure 37 shows the mechanical dimensions for the A/D converter card.

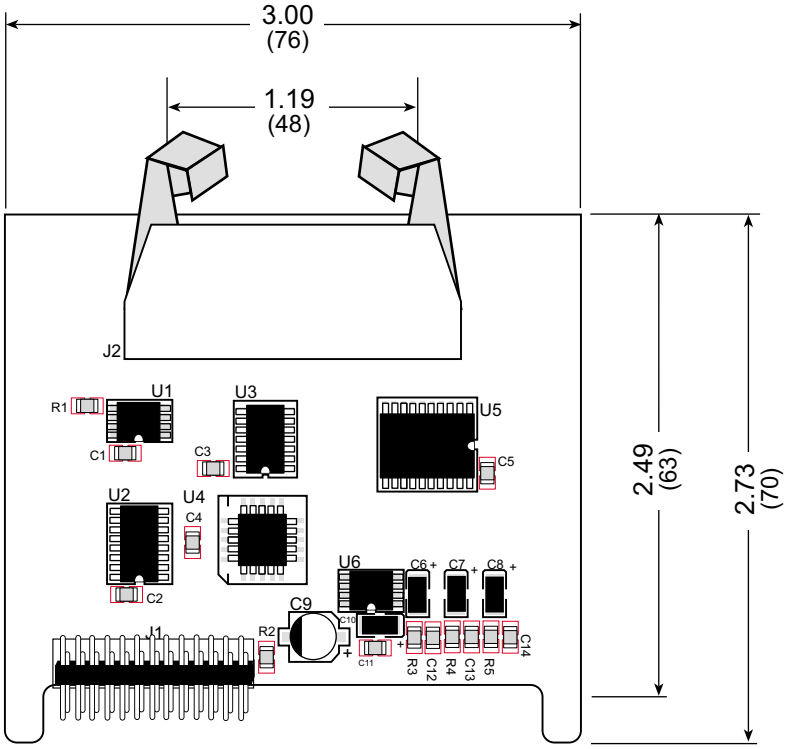


Figure 37. Relay Card Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

Table 14 lists the electrical, mechanical, and environmental specifications for the A/D converter card.

Table 14. A/D Converter Card Specifications

Parameter	Specification
Board Size	2.73" × 3.00" × 0.44" (70 mm × 76 mm × 11 mm)
Connectors	one 2 × 10 latch/eject ribbon connector, 0.1 inch pitch
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Power Requirements	5 V DC at 40 mA from backplane (+5 V supply) 9 V to 30 V DC, 35 mA at 24 V DC, **+RAW/+V_USER** from backplane
Number of Inputs	11 conditioned channels
Analog Input Ranges*	0 V to +10 V (max. ±22 V DC) –10 V to +10 V (max. ±40 V DC) 4 mA to 20 mA (max. 30 mA)
Resolution	12 bits (0–4095)
Conversion Time (including Dynamic C)	0.13 ms/channel (includes 0.08 ms/channel for raw count)
Repeatability	Typical ±½ count, maximum ±1 count @ –20°C to +70°C Typical ±1 count, maximum ±2 counts @ –40°C to –20°C
Accuracy	Typical ±1 count, maximum ±2 counts @ 25°C ±4 counts @ –40°C and +70°C†
Input Impedance	SR9300 (0 V to +10 V): 100 kΩ min. SR9310 (–10 V to +10 V): 100 kΩ min. SR9320 (4 mA to 20 mA): 249 Ω ± 1%
Linearity Error (end to end)	±1 count

* The A/D converter card is protected against transients that might exceed the maximum ratings.

† Accuracy at temperature extremes can be improved by recalibrating the A/D converter card at the temperature it will be used at.

PART IV. D/A CONVERTER CARDS

9. D/A CONVERTER CARDS

Chapter 9 describes the features of the D/A converter card, one of the I/O cards designed for the Smart Star embedded control system.

The Smart Star is a modular and expandable embedded control system whose configuration of I/O, A/D converter, D/A converter, and relay cards can be tailored to a large variety of demanding real-time control and data acquisition applications.

The typical Smart Star system consists of a rugged backplane with a power supply, a CPU card, and one or more I/O cards. The CPU card plugs into a designated slot on the backplane chassis, which has seven additional slots available for I/O cards to be used in any combination. A high-performance Rabbit 2000 microprocessor on the CPU card operates at 25.8 MHz to provide fast data processing.

9.1 D/A Converter Card Features

Three models of D/A converter cards are available, as shown in Table 15.

Table 15. Smart Star D/A Converter Cards

I/O Card	Model	Features
D/A Converter	SR9400	12-bit D/A converter, 8 channels, 0 V – 10 V
	SR9410	12-bit D/A converter, 8 channels, -10 V – +10 V
	SR9420	12-bit D/A converter, 8 channels, 4 mA – 20 mA

9.2 User Interface

Figure 38 shows the D/A converter circuit. A buffer, U6, buffers the data signals D0–D7 from the Smart Star backplane, and sends them to the D/A converter, U2–U5. Signals D2–D5 are used to switch the chip select line to identify which D/A converter will perform the conversion. The model of D/A converter card determines the analog output ranges (0 V to 10 V, -10 V to +10 V, or 4–20 mA). The different voltage or current ranges are handled with different feedback resistors, as shown in Figure 38. A switching regulator provides a regulated power supply for the op-amps.

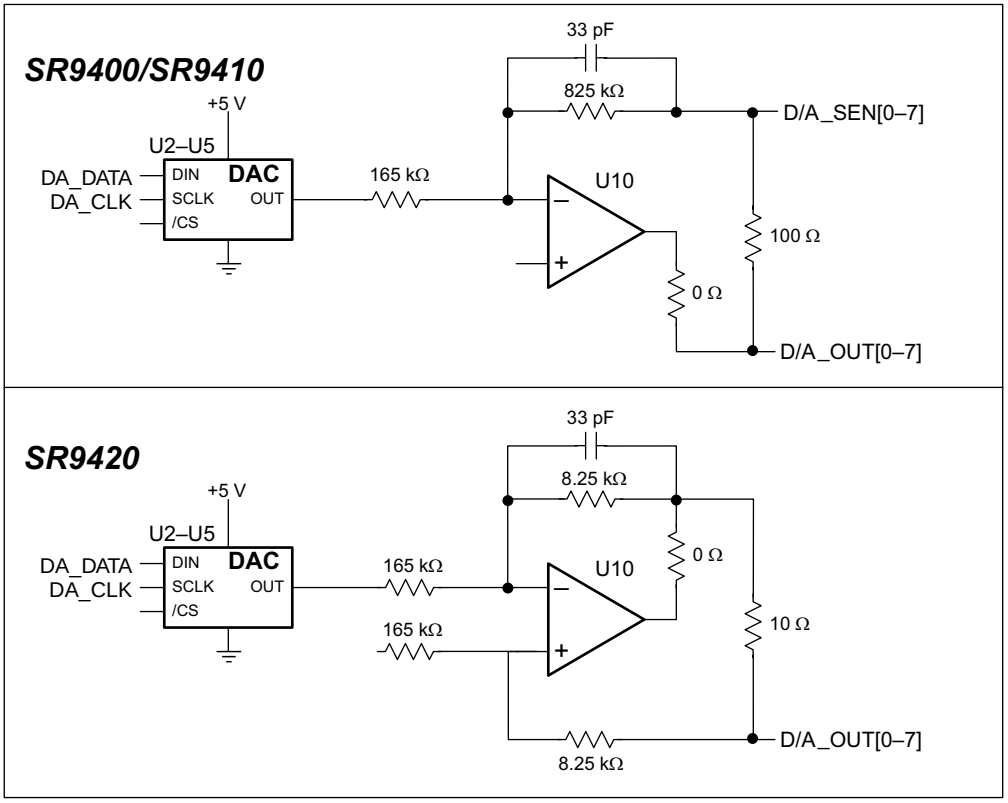


Figure 38. D/A Converter Card Circuit

NOTE: The **D/A_SEN[0–7]** sensing inputs are not used when using the current source version (model SR9420) of the D/A converter card.

Figure 39 shows the complete pinout for the user interface on header J1. Note that pin 1 is indicated by a small arrow on the ribbon cable connector.

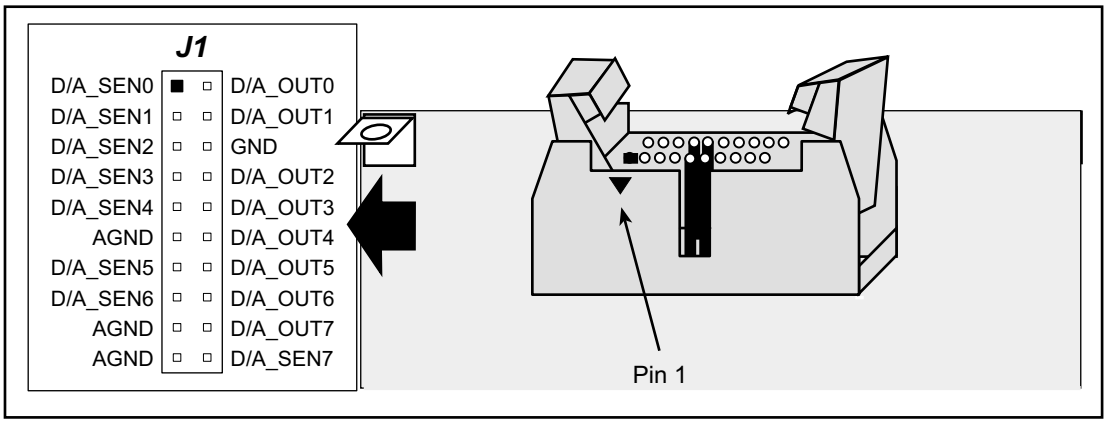


Figure 39. D/A Converter Card User Interface Pinout

The D/A converter card has eight analog output channels, **D/A_OUT[0–7]**, and is also equipped with a remote sensing capability through sensing inputs **D/A_SEN[0–7]** for the voltage-amplifier versions of the D/A converter card (models SR9400 and SR9410). These sensing inputs compensate for the voltage drop across the wire leads of low-impedance loads to provide a more precise output across the load.

Let's look at Figure 40 to see how this happens. Assume the load is 500 Ω. If the impedance of the wire used to connect the load to the output terminal on the D/A converter card is 5 Ω, there will be a voltage drop of about 5 Ω/500 Ω = 1% across the wire. The voltage across the load will then be 1% less, which is about 40 counts for the SR9400. By connecting **D/A_SEN** as shown in Figure 40, the output driver will be able to sense the voltage drop across the wire and provide a more accurate voltage output across the load. If the load impedance is much greater than the impedance of the wire leads, simply leave the **D/A_SEN** sensing inputs open.

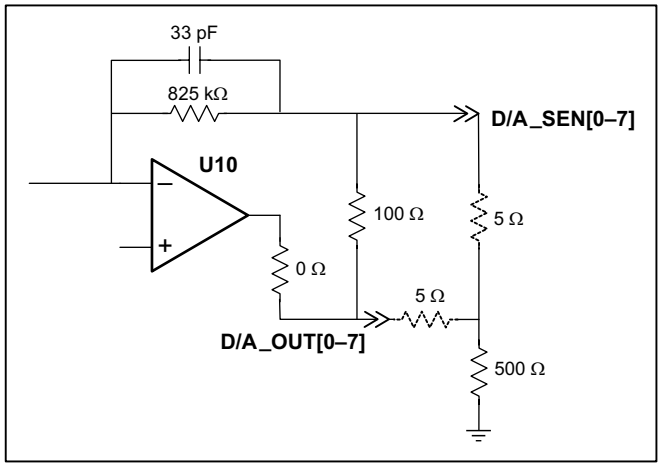


Figure 40. D/A Converter Output for Low-Impedance Loads

9.3 User FWT Connections

Connections to the D/A converter cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Table 16 lists the the Z-World part numbers for the FWTs.

Table 16. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals	Screw Terminals
			
FWT18	D/A Converter	101-0421	101-0425

9.3.1 Pinouts

Figure 41 shows the pinout for the FWTs used on the D/A converter cards.

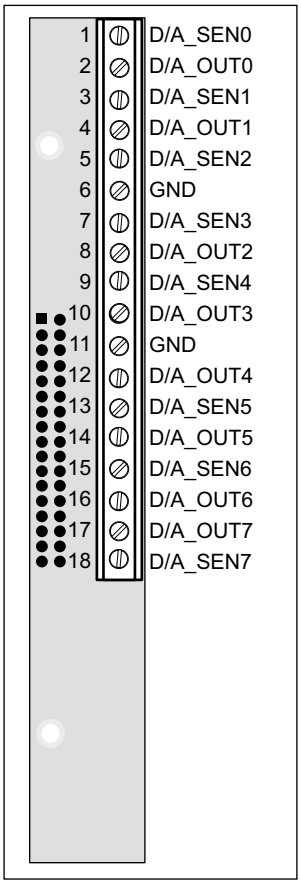


Figure 41. FWT Pinout for D/A Converter Cards

9.4 Power Distribution

Figure 42 shows the power distribution on the D/A converter card.

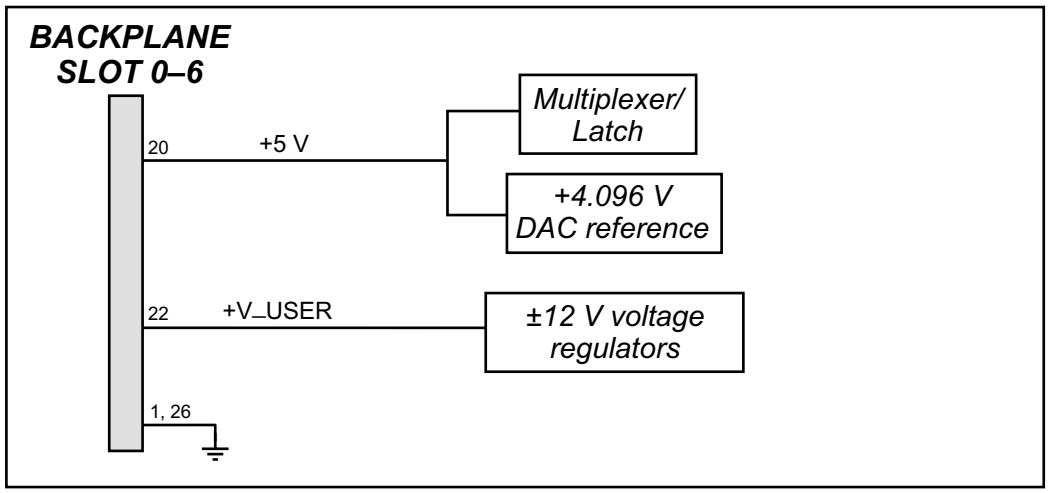


Figure 42. D/A Converter Card Power Distribution

Figure 43 shows the power supply for the op-amps used as voltage amplifiers/current sources.

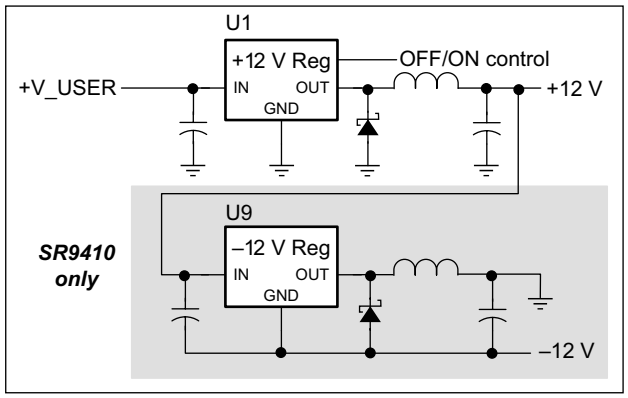


Figure 43. Op-Amp Power Supplies

There is provision in software using the **anaOutDisable** or the **anaOutEnable** function calls to turn the regulated ±12 V power supply off or on since pin 5 on U1 is connected to PE7 on the Rabbit 2000 microprocessor on the backplane. This type of disabling/enabling allows the analog output channels to float in a high-impedance state.

The voltage regulator on/off is disabled by default when there is a reset or when the D/A converter card is first used. All output channels must be configured to the required voltage or current outputs before calling the **anaOutEnable** function since unconfigured channels are automatically set to the maximum output.

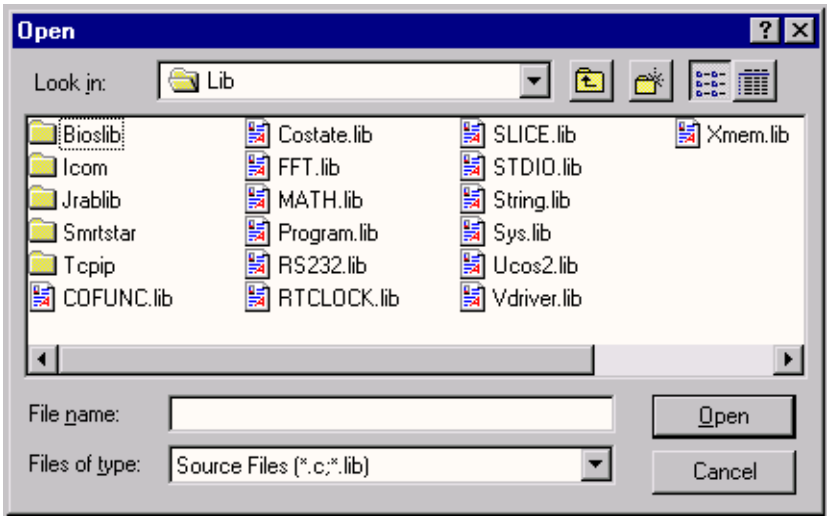
The –12 V supply is provided only for the SR9410, which provides analog outputs up to ±10 V.

9.5 Software

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

9.5.1 Dynamic C Libraries

With Dynamic C running, click **File > Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.

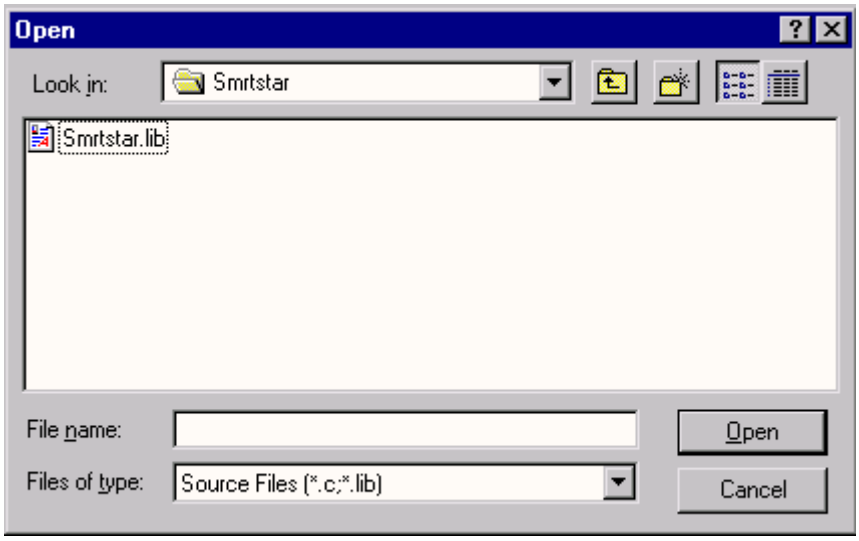


One library directory is specific to the Smart Star.

- **SMRTSTAR**—libraries associated with features specific to the Smart Star control system.

9.5.2 Library Directories

The **SMRTSTAR** directory contains libraries required to operate the Smart Star control system.



- **SMRTSTAR.LIB**—This library supports all the functions needed by the Smart Star systems including digital I/O cards, relay cards, A/D converter and D/A converter cards, and serial communication.

9.5.3 Smart Star D/A Converter Card Function APIs

void anaOutDisable(void);

Turns off (disables) voltage regulator for output-channel op-amps on ***all*** D/A converter cards, leaving all output channels in a high-impedance state.

RETURN VALUE

None.

See Also

anaOutEnable, anaOut, anaOutVolts, anaOutmAmps

void anaOutEnable(void);

Turns on (enables) voltage regulator for output-channel op-amps on ***all*** D/A converter cards.

NOTE: The voltage regulator on/off is disabled (off) at power-up or reset. All output channels must be configured to the required voltage or current outputs before calling the **anaOutEnable** function since unconfigured channels will be set automatically to the maximum output.

RETURN VALUE

None.

SEE ALSO

anaOutDisable, anaOut, anaOutVolts, anaOutmAmps

int anaOutEERd(int channel);

The D/A converter card calibration constants, gain, and offset are stored in the factory in the upper half of the EEPROM on the D/A converter card. Use this function to read the D/A converter card calibration constants into the global table **_dacCalib**

Parameters

channel is the D/A converter output channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7.

RETURN VALUE

0 if successful.
–1—control command unacceptable.
–2—EEPROM address unacceptable.

SEE ALSO

anaOutEEWr

```
int anaOutCalib(int channel, int value1,
    float voltamp1, int value2, float voltamp2);
```

Calibrates the response of the desired D/A converter channel as a linear function using the two conversion points provided. Gain and offset constants are calculated and placed into global table **_dacCalib**.

Parameters

channel is the D/A converter output channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7.

value1 is the first D/A conversion data point. Use a value near 4095 to produce a lower output measurement.

voltamp1 is the voltage (volts) or current (milliamperes) measurement corresponding to the first D/A conversion data point.

value2 is the second D/A conversion data point. Use a value near 0 to produce a higher output measurement.

voltamp2 is the voltage (volts) or current (milliamperes) corresponding to the second D/A conversion data point.

rawcount	Approximate Output Equivalent		
	SR9400	SR9410	SR9420
0 (0000H)	+10 V	+10 V	20 mA
2047 (07FFH)	+5 V	0 V	12 mA
4095 (0FFFH)	0 V	–10 V	4 mA

RETURN VALUE

0 if successful.
–1 if not able to make calibration constants.

SEE ALSO

anaOut, anaOutVolts

int anaSaveCalib(int boardtype);

The calibration constants may also be saved in the flash memory on the Smart Star CPU card. Doing so will speed up D/A conversions since a memory access from flash memory will be faster than from EEPROM. Use **anaSaveCalib** to save the current set of calibration constants for the analog input or output channels in the Smart Star flash memory. The calibration constants stored in flash memory can then be accessed at any time with the **anaLoadCalib** function.

Calibration constants should first be established using **anaOutCalib** or obtained via **anaOutEERd**.

PARAMETER

boardtype is the type of board, which is 0 for the D/A converter card, 1 for the A/D converter card.

RETURN VALUE

0 if successful.
–1—attempt to write non-flash area, nothing written.
–2—**rootSrc** not in root.
–3—timeout while writing flash memory.
–4—attempt to write to ID block sector(s).

SEE ALSO

anaLoadCalib, anaOutCalib

int anaLoadCalib(int boardtype);

Reads a complete set of calibration constants for the analog output channels from the Smart Star flash memory on the CPU card. These should have been loaded to the flash memory with the **anaSaveCalib** function.

PARAMETER

boardtype is the type of board, which is 0 for the D/A converter card, 1 for the A/D converter card.

RETURN VALUE

0 if successful.
–1—attempt to read from non-flash area.
–2—destination not all in root.

SEE ALSO

anaSaveCalib, anaOutCalib

```
int anaOut(unsigned int channel,
           unsigned int rawcount);
```

Sets the voltage of an analog output channel by serially clocking in 16 bits to a D/A converter using the following format:

- Program bits (D15…D12)
- New data (D11…D0)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R1	SPD	PWR	R0	MSB	12 data bits MSB–LSB (0–4095)										LSB

SPD—Speed control bit: 1 = fast mode (default), 0 = slow mode

PWR—Power control bit: 1 = power down, 0 = normal operation (default)

The following table lists all the possible combinations of the register-selects bits R1 (Register 1) and R0 (Register 0)

R1	R0	Register
0	0	Write data to D/A converter channel B
0	1	Write data to buffer
1	0	Write data to D/A converter channel A
1	1	Reserved

PARAMETERS

`channel` is the D/A converter output channel to write. **`channel`** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **`slotnumber`** is 0–6, and **`channelnumber`** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **`slotnumber`** is 0–6, and **`channelnumber`** is 0–7.

`rawcount` is a value corresponding to the voltage on the analog output channel (0–4095). The following **`rawcount`** data correspond to the analog outputs indicated.

`rawcount`	Approximate Output Equivalent		
	SR9400	SR9410	SR9420
0 (0000H)	+10 V	+10 V	20 mA
2047 (07FFH)	+5 V	0 V	12 mA
4095 (0FFFH)	0 V	–10 V	4 mA

RETURN VALUE

0 if successful.
–1 if **`rawcount`** is greater than 4095.

SEE ALSO

`anaOutVolts, anaOutCalib`

void anaOutVolts(unsigned int channel, float voltage);

Sets the voltage of an analog output channel by using the previously set calibration constants to calculate correct data values.

PARAMETERS

channel is the D/A converter output channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7.

voltage is the voltage desired on the output channel.

RETURN VALUE

None.

SEE ALSO

anaOut, anaOutCalib, anaOutmAmps

void anaOutmAmps(unsigned int channel, float current);

Sets the current of an analog output channel by using the previously set calibration constants to calculate correct data values.

PARAMETERS

channel is the D/A converter output channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7.

voltage is the voltage desired on the output channel.

RETURN VALUE

0 if successful.

–1 if not able to make calibration constants.

SEE ALSO

anaOut, anaOutVolts, anaOutCalib

int anaOutEEWr(int channel);

Writes the calibration constants, gain, and offset to the upper half of the EEPROM on the D/A converter card.

PARAMETERS

channel is the D/A converter output channel. **channel** should be passed as

```
channel = (slotnumber * 128) + (channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7

or

```
channel = ChanAddr(slotnumber, channelnumber)
```

where **slotnumber** is 0–6, and **channelnumber** is 0–7.

voltage is the voltage desired on the output channel.

RETURN VALUE

0 if successful.
-1—control command unacceptable.
-2—EEPROM address unacceptable.
-3—data value unacceptable.

SEE ALSO

anaOutEERd

9.5.4 Sample Programs

- **ANAVOUT.C**—Demonstrates how to set the D/A channel for the desired output.
- **SSDAC1.C**—Demonstrates how to recalibrate a D/A converter channel using two known voltages, and shows how to define the two coefficients, gain and offset, that will be rewritten into the D/A converter card's EEPROM.
- **SSDAC2.C**—Demonstrates how to recalibrate a D/A converter channel using an A/D converter card andtwo known voltages. Shows how to define the two coefficients, gain and offset, that will be rewritten into the D/A converter card's EEPROM.
- **SSDAC3.C**—Demonstrates how to recalibrate a D/A converter channel using two known currents, and shows how to define the two coefficients, gain and offset, that will be rewritten into the D/A converter card's EEPROM.
- **SSDAC4.C**—Demonstrates how to recalibrate a D/A converter channel using an A/D converter card,two known currents. Shows how to define the two coefficients, gain and offset, that will be rewritten into the D/A converter card's EEPROM.

9.5.4.1 Using Dynamic C

To run a sample program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The CPU card must be in Program Mode (see Section 4.1, “Programming Cable”) and must be connected to a PC using the programming cable as described in Section 2.3, “Programming Cable Connections.”

More complete information on Dynamic C is provided in the ***Dynamic C Premier User’s Manual***.

9.6 Electrical and Mechanical Specifications

Figure 44 shows the mechanical dimensions for the D/A converter card.

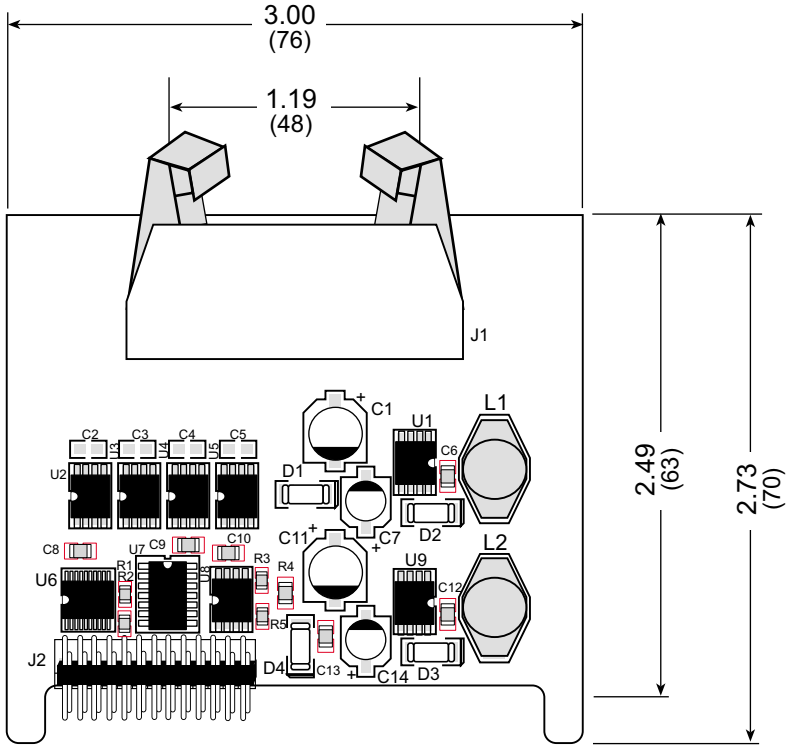


Figure 44. D/A Converter Card Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

Table 17 lists the electrical, mechanical, and environmental specifications for the D/A converter card.

Table 17. D/A Converter Card Specifications

Parameter	Specification
Board Size	2.73" × 3.00" × 0.44" (70 mm × 76 mm × 11 mm)
Connectors	one 2 × 10 latch/eject ribbon connector, 0.1 inch pitch
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Power Requirements	5 V DC at 50 mA typical from backplane (+5 V supply) 15 V to 30 V DC, 30 mA at 24 V DC, **+RAW/+V_USER** from backplane
Number of Outputs	8 channels
Analog Output Ranges	SR9400: 0 V to +10 V, 20 mA/channel (maximum) SR9410: –10 V to +10 V, 20 mA/channel (maximum) SR9420: 4 mA to 20 mA, 10 V (maximum)
Resolution	12 bits (0–4095)
Conversion Time (including Dynamic C)	0.2 ms/channel
Output Stability	±½ count
Output Impedance	SR9400: < 1 Ω, SR9410: < 1 Ω, SR9420: > 100 kΩ

PART V. RELAY CARDS

10. RELAY CARDS

Chapter 10 describes the features of the relay card, one of the I/O cards designed for the Smart Star embedded control system.

The Smart Star is a modular and expandable embedded control system whose configuration of I/O, A/D converter, D/A converter, and relay cards can be tailored to a large variety of demanding real-time control and data acquisition applications.

The typical Smart Star system consists of a rugged backplane with a power supply, a CPU card, and one or more I/O cards. The CPU card plugs into a designated slot on the backplane chassis, which has seven additional slots available for I/O cards to be used in any combination. A high-performance Rabbit 2000 microprocessor on the CPU card operates at 25.8 MHz to provide fast data processing.

10.1 Relay Card Features

Two models of relay cards are available, as shown in Table 18.

Table 18. Smart Star Relay Cards

I/O Card	Model	Features
Relay	SR9500	5 SPST relays and 1 SPDT relay, each protected with onboard snubbers
	SR9510	8 SPDT relays (no snubbers)

The SR9500 relay cards are suitable for switching all kinds of loads up to 30 V DC at 1 A or 48 V AC at 0.5 A. The SR9510 handles similar loads, but is restricted to noninductive loads unless you add snubbers to the system that is interfacing with the Smart Star.

10.2 User Interface

Depending on the model of relay card (see Table 18), the relays on the relay card will be configured as SPDT or SPST with or without snubbers. Figure 45 shows these relay configurations.

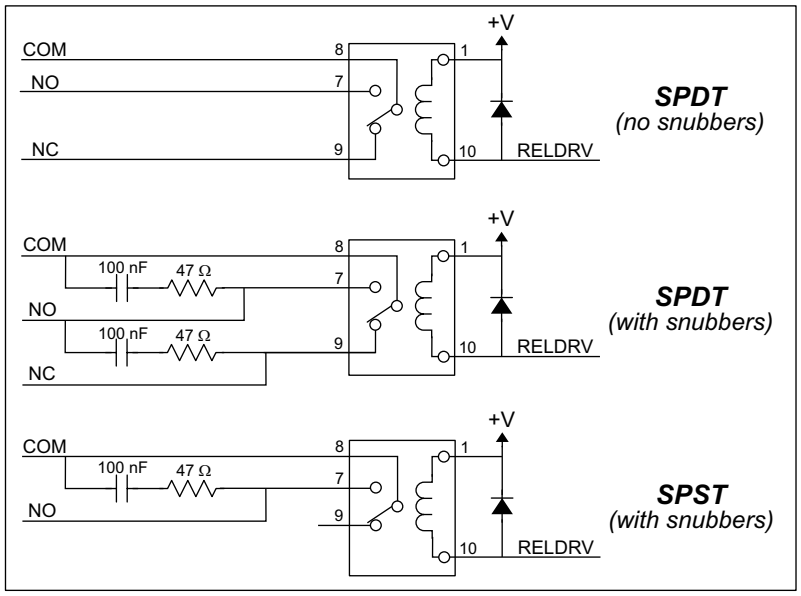


Figure 45. Relay Configurations

The diode protects the coil power supply (and the Smart Star backplane) from inductive spikes caused by energizing/de-energizing the coil, and the resistor-capacitor snubbers protect the relay contacts against voltage spikes induced by inductive loads.

Figure 46 shows the complete pinout for the user interface on header J1. Note that pin 1 is indicated by a small arrow on the ribbon cable connector.

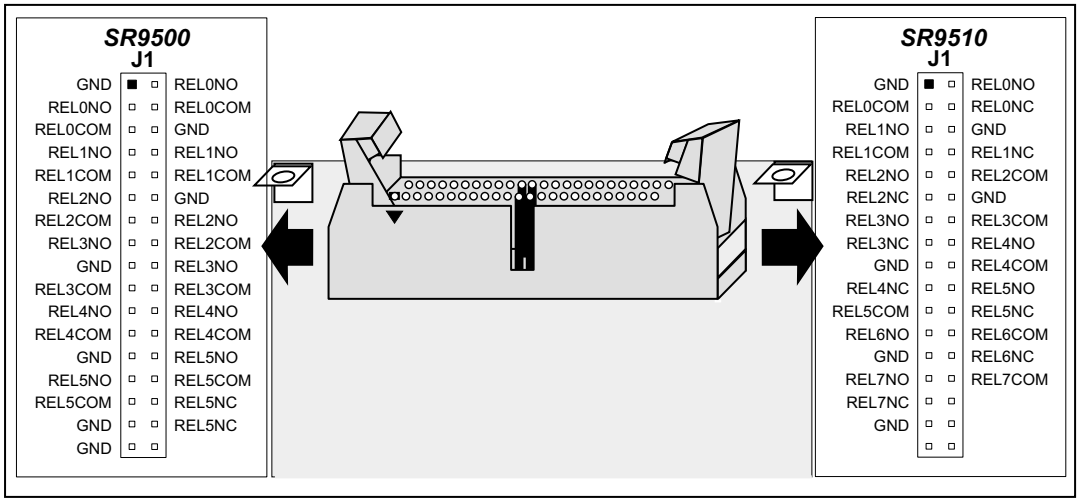


Figure 46. Relay Card User Interface Pinout

10.3 User FWT Connections

Connections to the relay cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Table 19 lists the Z-World part numbers for the FWTs.

Table 19. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals	Screw Terminals
			
FWT18R	Relay (SR9500)	101-0422	101-0426
FWT27	Relay (SR9510)	101-0420	101-0424

10.3.1 Pinouts

Figure 47 shows the pinout for the FWTs used on the relay cards.

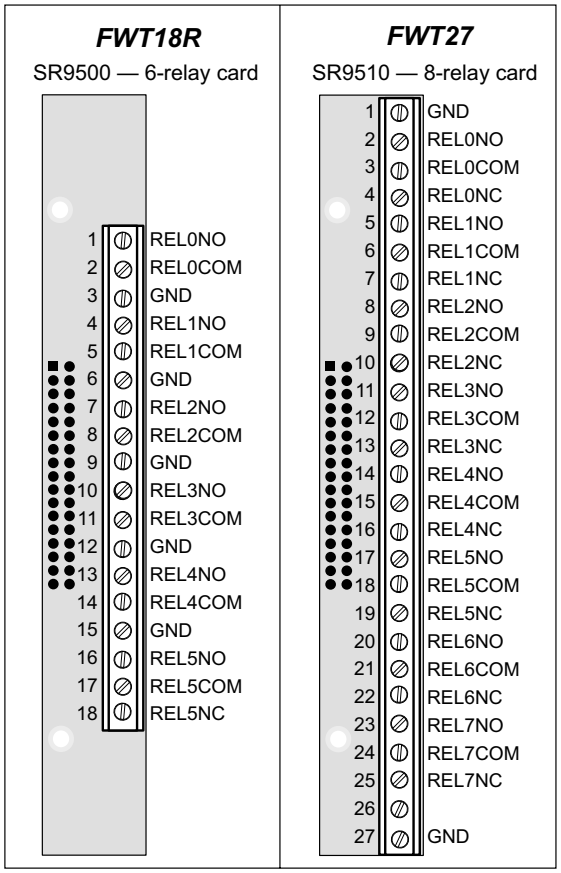


Figure 47. FWT Pinouts for Relay Cards

10.4 Power Distribution

Figure 48 shows the power distribution on the relay card.

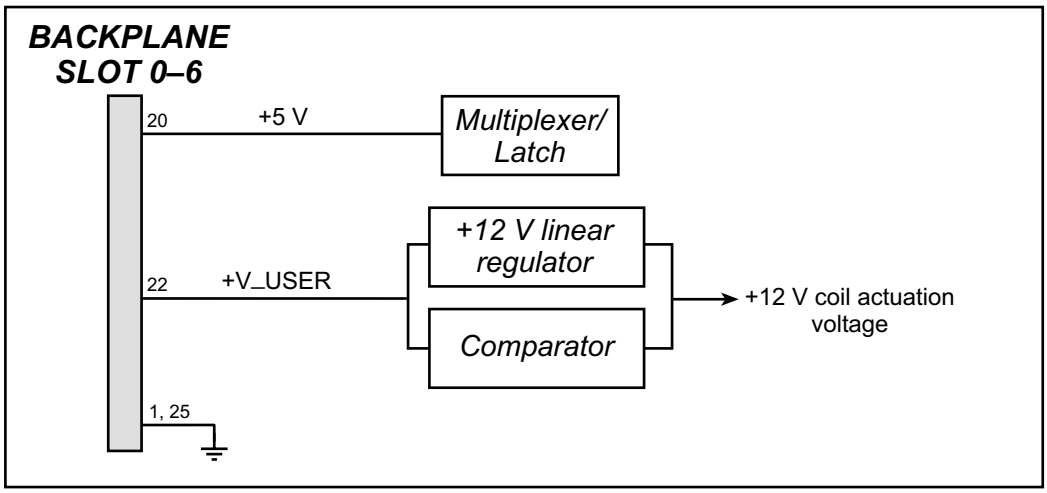


Figure 48. Relay Card Power Distribution

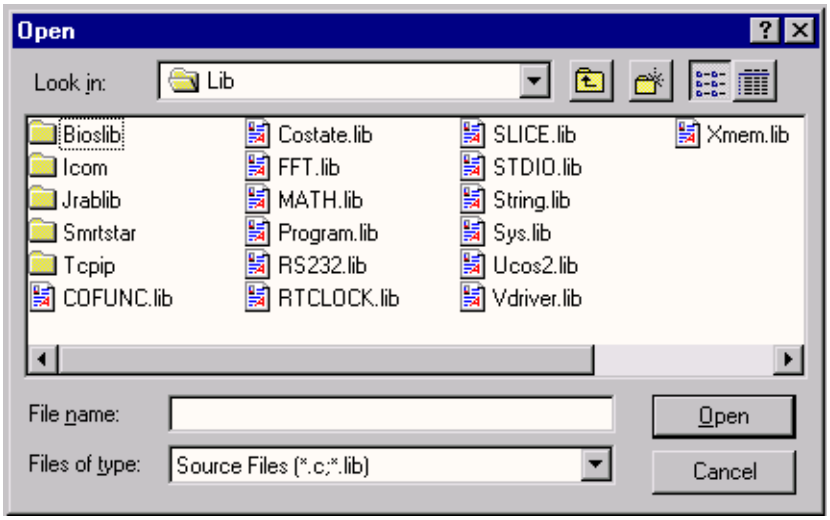
The relay coil actuation voltage is 12 V, and so **+V_USER** should be 12 V to 30 V DC. The **+V_USER** supply passes through a linear regulator and comparator, which are in parallel. The comparator is set for approximately +13.9 V, and as long as **+V_USER** is more than +13.9 V, the +12 V from the linear regulator will provide the coil actuation voltage. Should **+V_USER** be less than +13.9 V, the comparator will supply **+V_USER** directly to provide the coil actuation voltage.

10.5 Relay Cards Software

Dynamic C Premier is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Z-World controllers and other controllers based on the Rabbit microprocessor.

10.5.1 Dynamic C Libraries

With Dynamic C running, click **File > Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.

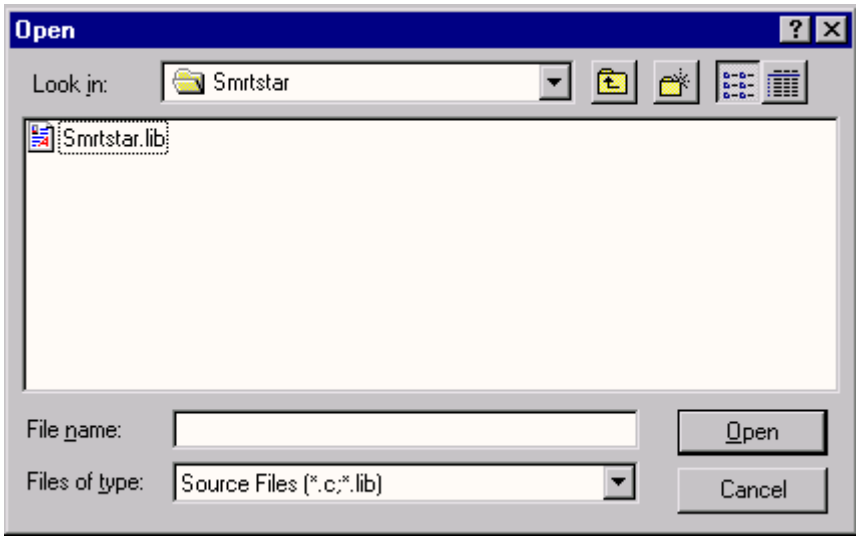


One library directory is specific to the Smart Star.

- **SMRTSTAR**—libraries associated with features specific to the Smart Star control system.

10.5.2 Library Directories

The **SMRTSTAR** directory contains libraries required to operate the Smart Star control system.



- **SMRTSTAR.LIB**—This library supports all the functions needed by the Smart Star systems including digital I/O cards, relay cards, D/A converter and A/D converter cards, and serial communication.

10.5.3 Smart Star Relay Card Function APIs

```
void relayOut(int relay, int value);
```

Sets the state of a relay.

PARAMETER

relay is the relay to set. **relay** should be passed as

```
relay = (slotnumber * 128) + (relaynumber)
```

or

```
relay = ChanAddr(slotnumber, relaynumber)
```

where **slotnumber** is 0–6, and **relaynumber** is 0–5 (SR9500) or 0–7 (SR9510), depending on the model of relay card.

value is the value to set the relay to, 0 or 1 (off or on).

10.5.4 Sample Programs

- **SSTARRLY.C**—Demonstrates turning a relay on the relay card on and off.

10.6 Using Dynamic C

To run a sample program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The CPU card must be in Program Mode (see Section 4.1, "Programming Cable") and must be connected to a PC using the programming cable as described in Section 2.3, "Programming Cable Connections."

More complete information on Dynamic C is provided in the ***Dynamic C Premier User's Manual***.

10.7 Electrical and Mechanical Specifications

Figure 49 shows the mechanical dimensions for the relay card.

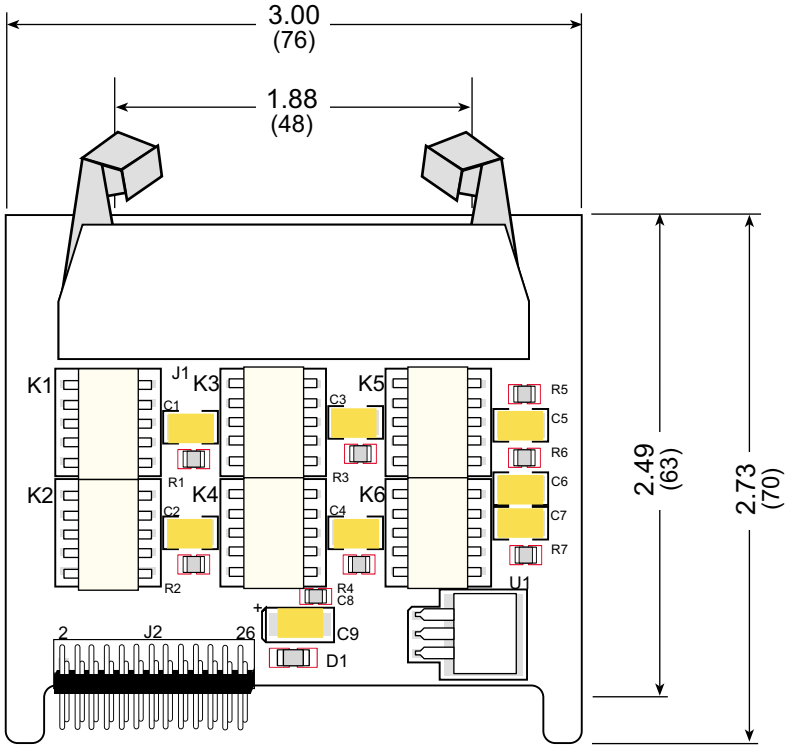


Figure 49. Relay Card Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

Table 20 lists the electrical, mechanical, and environmental specifications for the relay card.

Table 20. Relay Card Specifications

Parameter	Specification
Board Size	2.73" × 3.00" × 0.44" (70 mm × 76 mm × 11 mm)
Connectors	one 2 × 17 latch/eject ribbon connector, 0.1 inch pitch
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Power Requirements	5 V DC at 10 mA from backplane (+5 V supply) 12 V to 30 V DC, 10 mA at 24 V DC, **+RAW/+V_USER** from backplane
Relay Switching Contacts	30 V DC at 1 A or 48 V AC at 0.5 A
Relays	SR9500: 1 SPDT, 5 SPST (N.O., COM) with snubbers SR9510: 8 SPDT (N.O., N.C., COM), no snubbers

PART VI. APPENDICES

APPENDIX A.
FIELD WIRING TERMINALS

Appendix A explains how to prepare the connector on an I/O card to accept a field wiring terminal, and how to secure the field wiring terminal to the I/O card. The dimensions for the field wiring terminals are included.

A.1 Selecting and Installing a Field Wiring Terminal

Connections to the I/O cards are made via a ribbon cable connector or optional field wiring terminals that are either pluggable or have screw terminals. Three different Field Wiring Terminals (FWTs) are available. Table A-1 lists the I/O cards and the Z-World part numbers for the corresponding FWTs.

Table A-1. Guide to FWT Selection

FWT Description	I/O Cards	Z-World Part Number	
		Pluggable Terminals 	Screw Terminals 
FWT27	Digital I/O (SR9200 series)	101-0420	101-0424
FWT18	A/D Converter (SR9300 series) D/A Converter (SR9400 series) Relay (SR9510)	101-0421	101-0425
FWT18R	Relay (SR9500)	101-0422	101-0426

Before you can install the FWT you selected for your I/O card, you must remove the tabs from the connector on the I/O card. To do so, move the tab inwards as far as possible, as shown in Figure A-1. Then insert a screwdriver into the space below the tab on the side of the connector and gently nudge the tab up and out. If you are careful, the tab will remain intact to be saved and snapped back in place should you need to use a ribbon cable connector in the future.

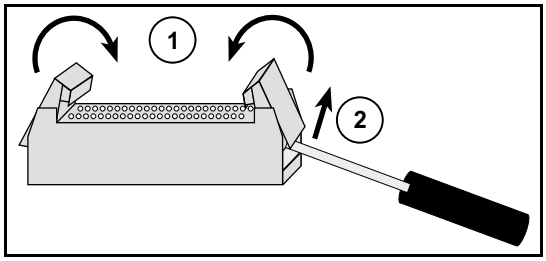


Figure A-1. Remove Tabs from Connector on I/O Card

Plug the FWT connector into the connector on the I/O card. Be sure to position the pluggable or screw connectors so that the edge of the FWT they are on faces outwards from the I/O card as shown in Figure A-2. Position the mylar insulator above the FWT as shown in Figure A-2 to protect the header pins on the printed circuit board, and secure the FWT using the two 4-40 × ¼ screws supplied.

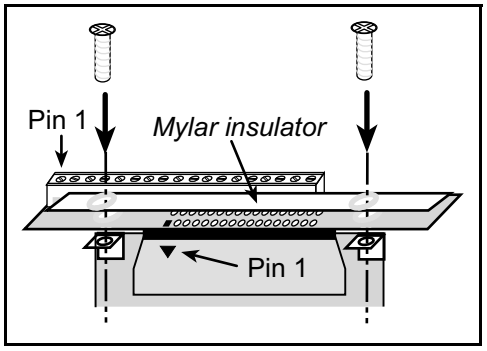


Figure A-2. Secure FWT to I/O Card

A.2 Dimensions

Figure A-3 shows the overall FWT dimensions.

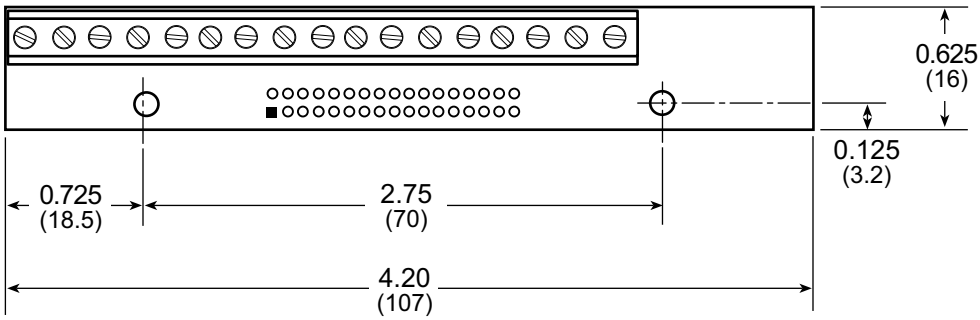


Figure A-3. FWT Dimensions

NOTE: All diagram and graphic measurements are in inches followed by millimeters enclosed in parentheses.

The actual appearance of the terminals may vary, depending on the number and type of terminals. The pinouts for the FWTs applicable to a particular I/O card are shown with the pinouts for the connectors on the individual I/O cards.

APPENDIX B. LCD/KEYPAD MODULE

An optional LCD/keypad module is available for the Smart Star. Appendix B describes the LCD/keypad module and provides the software APIs to make full use of the LCD/keypad module.

B.1 Specifications

Table B-1 lists the electrical, mechanical, and environmental specifications for the LCD/keypad module.

Table B-1. LCD/Keypad Specifications

Parameter	Specification
Board Size	2.60" × 3.00" × 0.75" (66 mm × 76 mm × 19 mm)
Temperature	Operating Range: 0°C to +50°C Storage Range: –40°C to +85°C
Humidity	5% to 95%, noncondensing
Power Consumption	1.5 W maximum
Connections	Connects to high-rise header sockets on Smart Star
LCD Panel Size	122 × 32 graphic display
Keypad	7-key keypad
LEDs	Seven user-programmable LEDs

Figure B-1 shows the mechanical dimensions for the LCD/keypad module and its bezel.

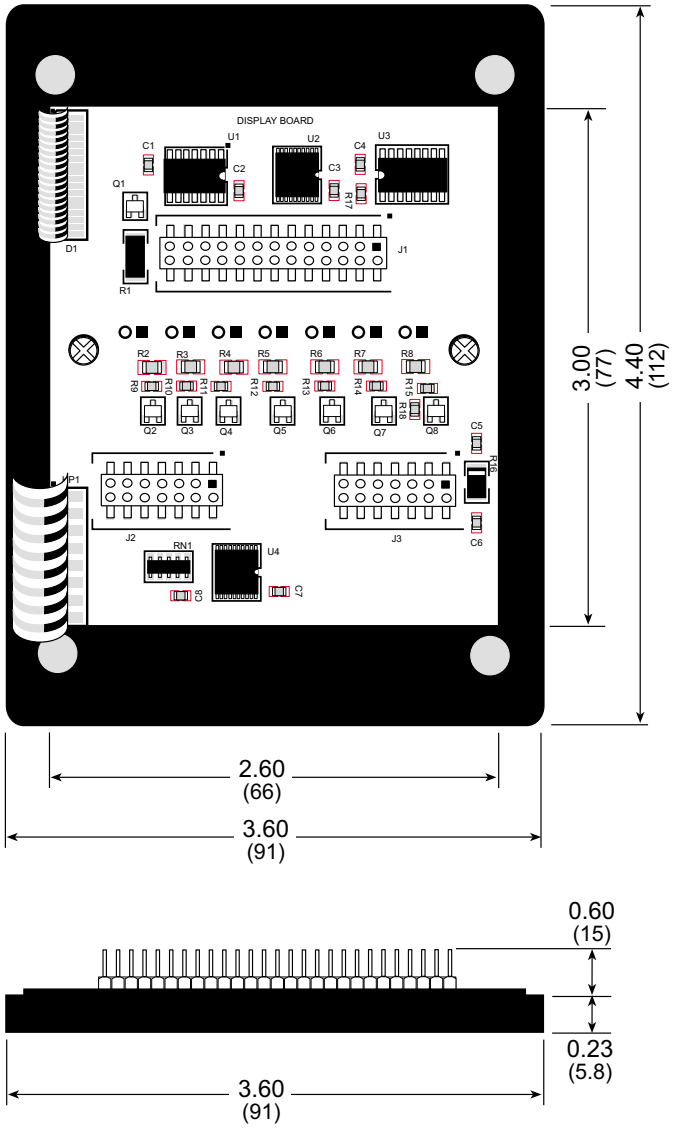


Figure B-1. LCD/Keypad Module and Bezel Dimensions

NOTE: All measurements are in inches followed by millimeters enclosed in parentheses.

B.2 Keypad Labeling

The keypad may be labeled according to your needs. A template is provided in Figure B-2 to allow you to design your own keypad label insert.

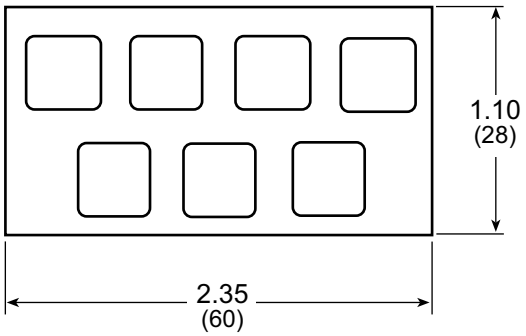


Figure B-2. Keypad Template

To replace the keypad legend, remove the old legend and insert your new legend prepared according to the template in Figure B-2. The keypad legend is located under the blue keypad matte, and is accessible from the left only as shown in Figure B-3.

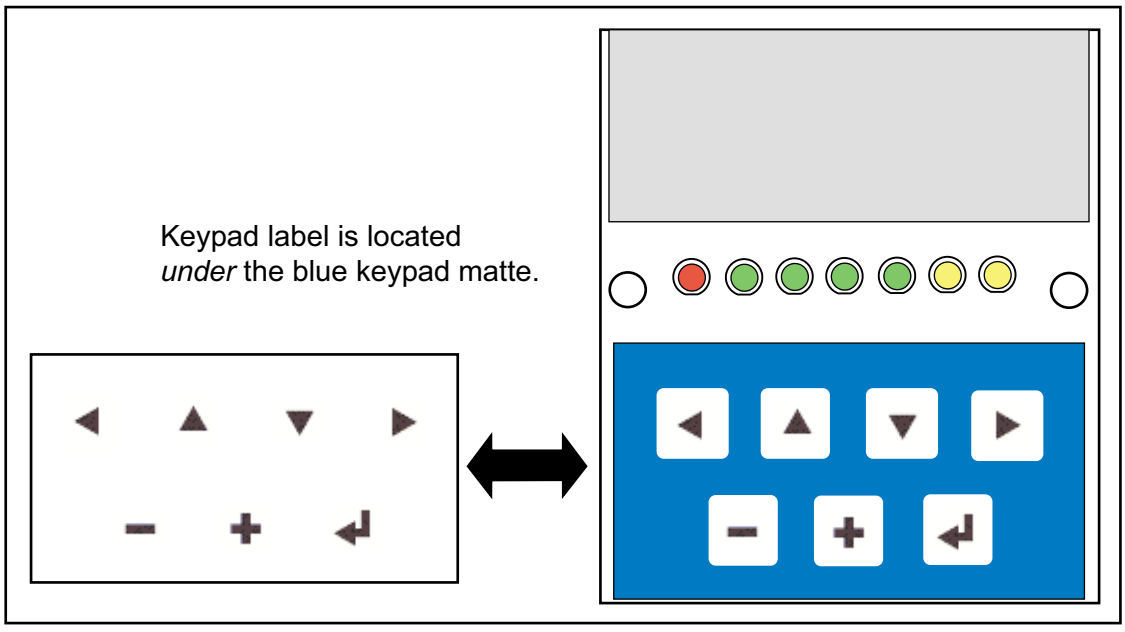


Figure B-3. Removing and Inserting Keypad Label

B.3 Header Pinouts

Figure B-4 shows the pinouts for the LCD/keypad module.

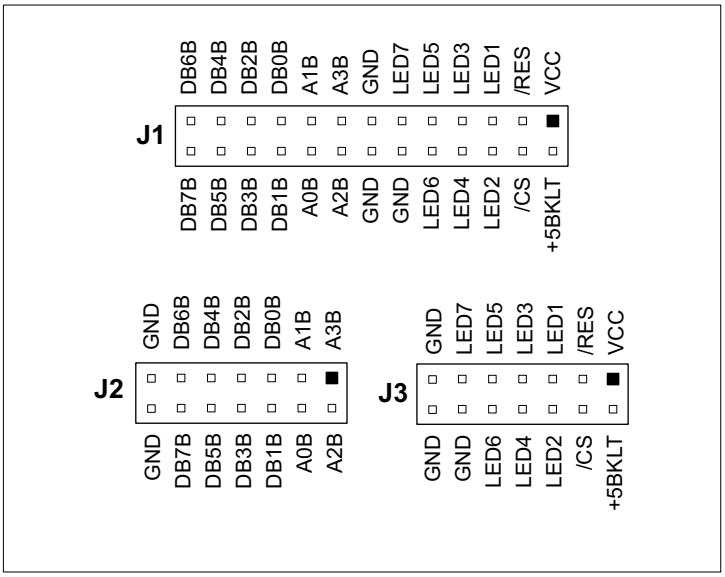


Figure B-4. LCD/Keypad Module Pinouts

B.3.1 I/O Address Assignments

The LCD and keypad on the LCD/keypad module are addressed by the /CS chip select as explained in Table B-2.

Table B-2. LCD/Keypad Module Address Assignment

Address	Function
61C0Exx0–61C0Exx7	LCD control
61C0Exx8	LED enable
61C0Exx9	Not used
61C0ExxA	7-key keypad
61C0ExxB (bits 0–6)	7-LED driver
61C0ExxB (bit 7)	LCD backlight on/off
61C0ExxC–61C0ExxF	Not used

B.4 Mounting LCD/Keypad Module

B.4.1 Installation Guidelines

When possible, following these guidelines when mounting the LCD/keypad module.

1. Leave sufficient ventilation space
2. Do not install the LCD/keypad module directly above machinery that radiates a lot of heat (for example, heaters, transformers, and high-power resistors).
3. Leave at least 8" (20 cm) distance from electric power lines and even more from high-voltage devices.
4. When installing the LCD/keypad module near devices with strong electrical or magnetic fields (such as solenoids), allow a least 3" (8 cm), more if necessary.

The LCD/keypad module has strong environmental resistance and high reliability, but you can maximize system reliability by avoiding or eliminating the following conditions at the installation site.

- Abrupt temperature changes and condensation
- Ambient temperatures exceeding a range of 0°C to 50°C
- Relative humidity exceeding a range of 5% to 95%
- Strong magnetism or high voltage
- Corrosive gases
- Direct vibration or shock
- Excessive iron dust or salt
- Spray from harsh chemicals

B.4.2 Mounting Instructions

A bezel and a gasket are included with the LCD/keypad module. When properly mounted in a panel, the LCD/keypad module bezel is designed to meet NEMA 4 specifications for water resistance.

Since the LCD/keypad module employs an LCD display, the viewing angle must be considered when mounting the display. Install the LCD/keypad module at a height and angle that makes it easy for the operator to see the screen.

B.4.2.1 Bezel-Mount Installation

This section describes and illustrates how to bezel-mount the LCD/keypad module. Follow these steps for bezel-mount installation.

1. Cut mounting holes in the mounting panel in accordance with the recommended dimensions in Figure B-5, then use the bezel faceplate to mount the LCD/keypad module onto the panel.

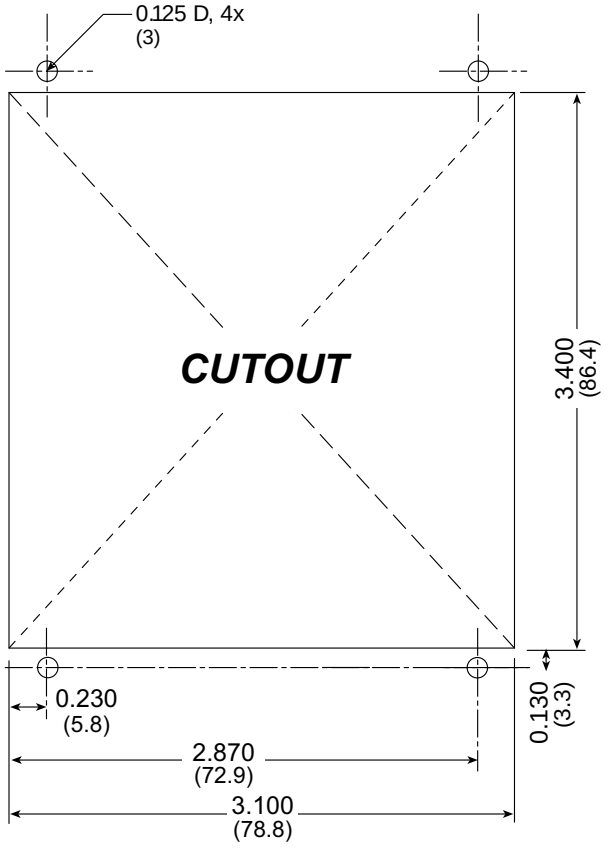


Figure B-5. Recommended Cutout Dimensions

2. Carefully “drop in” the LCD/keypad module with the bezel and gasket attached.

3. Fasten the unit with the four 4-40 screws and washers included with the LCD/keypad module. If your panel is thick, use a 4-40 screw that is approximately 3/16" (5 mm) longer than the thickness of the panel.

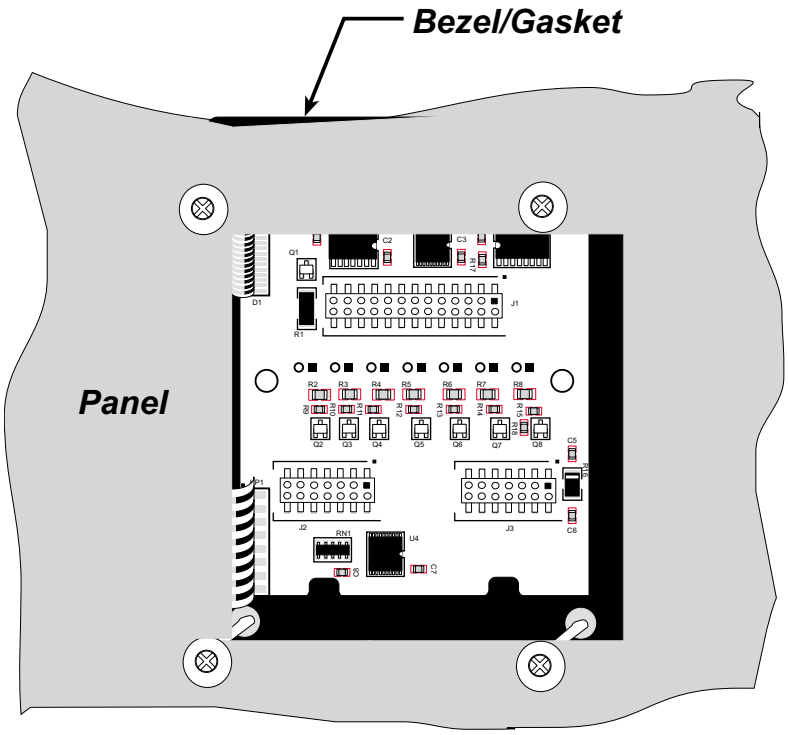


Figure B-6. LCD/Keypad Module Mounted in Panel (rear view)

Carefully tighten the screws until the gasket is compressed and the plastic bezel face-plate is touching the panel.

Do not tighten each screw fully before moving on to the next screw. Apply only one or two turns to each screw in sequence until all are tightened manually as far as they can be so that the gasket is compressed and the plastic bezel faceplate is touching the panel.

B.5 Connecting LCD/Keypad Module to Smart Star Backplane

The LCD/keypad module can be located as far as 2 ft. (60 cm) away from the Smart Star backplane, and is connected via a ribbon cable as shown in Figure B-7.

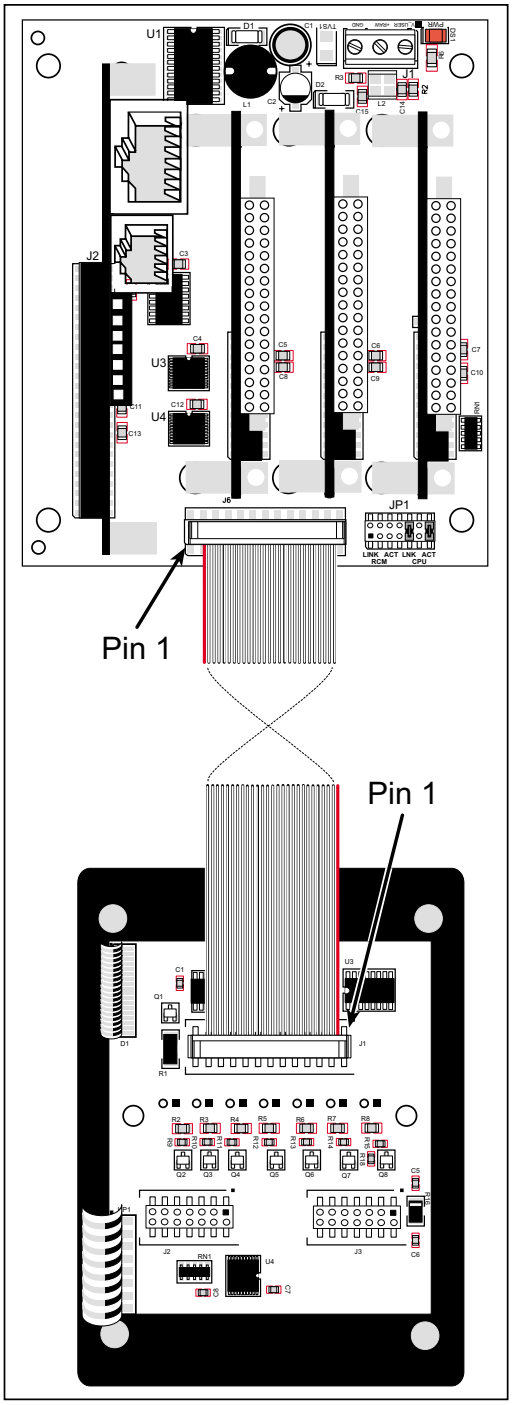


Figure B-7. Connecting LCD/Keypad Module to Backplane

Note the locations and connections for pin 1 on both the backplane and the LCD/keypad module.

The SR9050 backplane can also be panel-mounted behind the LCD/keypad module.

1. Prepare a cutout and install the LCD/keypad module in the cutout as explained in Section B.4.2.1.
2. Use brackets to secure the LCD/keypad module to the panel using the four 4-40 screws and washers included with the LCD/keypad module. The four screw positions are indicated with the number 1 in Figure B-8. If your panel is thick, use a 4-40 screw that is approximately 3/16" (5 mm) longer than the thickness of the panel.
3. Use a ribbon cable to connect header J6 on the backplane to header J1 on the LCD/keypad module. Note the pin 1 positions reflected by the red-colored line in the ribbon cable shown in Figure B-8.

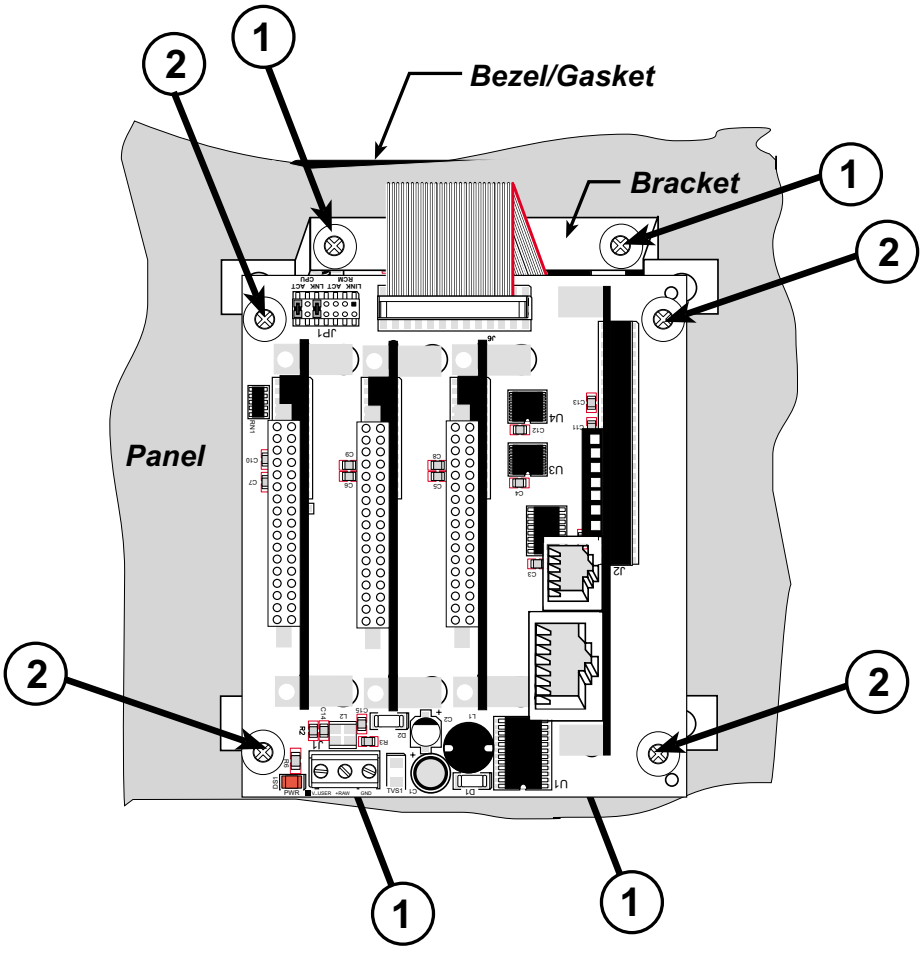


Figure B-8. Install Smart Star Backplane Behind LCD/Keypad Module

4. Secure the Smart Star backplane using four 4-40 × ½" or 6-32 × ½" screws at the screw positions indicated with the number 2 in Figure B-8.

Brackets and ribbon cables are sold separately. Note that only a Smart Star assembly using the SR9050 backplane can be panel-mounted.

B.6 Font and Bitmap Converter

A ***Font and Bitmap Converter*** tool is available to convert Windows fonts and monochrome bitmaps to a library file format compatible with Z-World's Dynamic C applications and graphical displays. Non-Roman characters can also be converted by applying the monochrome bitmap converter to their bitmaps.

Start the ***Font and Bitmap Converter*** tool by double-clicking on the **fbmcnvtr.exe** file in the Dynamic C directory. You then select and convert existing fonts or bitmaps. Complete instructions are available via the **Help** menu that is in the ***Font and Bitmap Converter*** tool.

Once you are done, the converted file is displayed in the editing window. Editing may be done, but should not be necessary. Save the file as ***libraryfilename*.lib**, where ***libraryfilename*** is a file name of your choice.

Add the library file(s) to applications with the statement **#use *libraryfilename*.lib,** or by cutting and pasting from the library file(s) you created into the application program.

> **TIP:** If you used the **#use *libraryfilename*.lib** statement, remember to enter ***libraryfilename*.lib** into **lib.dir**, which is located in your Dynamic C directory.

You are now ready to add the font or bitmap to your application using the **glXFontInit** or the **glXPutBitmap** function calls.

B.7 LCD/Keypad Module Function APIs

B.7.1 LEDs

When power is applied to the LCD/keypad module for the first time, the red LED (DS1) will come on, indicating that power is being applied to the LCD/keypad module. The red LED is turned off when the **brdInit** function executes.

One function is available to control the LEDs, and can be found in the **SMRTSTAR.LIB** library.

```
void ledOut(int led, int value);
```

LED on/off control. This function will only work when the LCD/keypad module is installed on the Smart Star.

PARAMETERS

led is the LED to control.

0 = LED DS1
1 = LED DS2
2 = LED DS3
3 = LED DS4
4 = LED DS5
5 = LED DS6
6 = LED DS7

value is the value used to control whether the LED is on or off (0 or 1).

0 = off
1 = on

RETURN VALUE

None.

SEE ALSO

brdInit

B.7.2 LCD Display

The functions used to control the LCD display are contained in the **GRAPHIC.LIB** library located in the Dynamic C **DISPLAYS\GRAPHIC** library directory.

void glInit(void);

Initializes the display devices, clears the screen.

RETURN VALUE

None.

SEE ALSO

glDispOnOFF, glBacklight, glSetContrast, glPlotDot, glBlock, glPlotDot, glPlotPolygon, glPlotCircle, glHScroll, glVScroll, glXFontInit, glPrintf, glPutChar, glSetBrushType, glBuffLock, glBuffUnlock, glPlotLine

void glBackLight(int onOff);

Sets the intensity of the backlight, if circuitry is installed.

PARAMETER

: **onOff** reflects the low to high values (typically 0 to 255, depending on the board design) to set the backlight intensity (0 will turn the backlight off completely.)

RETURN VALUE

None.

SEE ALSO

glInit, glDispOnoff, glSetContrast

void glDispOnOff(int onOff);

Sets the LCD screen on or off. Data will not be cleared from the screen.

PARAMETER

onOff turns the LCD screen on or off

1—turn the LCD screen on
0—turn the LCD screen off

RETURN VALUE

None.

SEE ALSO

glInit, glSetContrast, glBackLight

void glSetContrast(unsigned level);

Sets display contrast (the circuitry is *not* installed on the LCD/keypad module used with the Smart Star).

PARAMETER

`level` reflects low to high values (typically 0 to 255, depending on the board design) to give high to low contrast respectively.

RETURN VALUE

None.

SEE ALSO

`glInit, glBacklight, glDispOnoff`

void glFillScreen(char pattern);

Fills the LCD display screen with a pattern.

PARAMETER

The screen will be set to all black if **`pattern`** is 0xFF, all white if **`pattern`** is 0x00, and vertical stripes for any other pattern.

RETURN VALUE

None.

SEE ALSO

`glBlock, glBlankScreen, glPlotPolygon, glPlotCircle`

void glBlankScreen(void);

Blanks the LCD display screen (sets LCD display screen to white).

RETURN VALUE

None.

SEE ALSO

`glFillScreen, glBlock, glPlotPolygon, glPlotCircle`

void glBlock(int x, int y, int bmWidth, int bmHeight);

Draws a rectangular block in the page buffer and on the LCD if the buffer is unlocked. Any portion of the block that is outside the LCD display area will be clipped.

PARAMETERS

x is the *x* coordinate of the upper left corner of the block.

y is the *y* coordinate of the left top corner of the block.

bmWidth is the width of the block.

bmWidth is the height of the block.

RETURN VALUE

None.

SEE ALSO

glFillScreen, glBlankScreen, glPlotPolygon, glPlotCircle

void glPlotVPolygon(int n, int *pFirstCoord);

Plots the outline of a polygon in the LCD page buffer, and on the LCD if the buffer is unlocked. Any portion of the polygon that is outside the LCD display area will be clipped. The function will also return, doing nothing, if there are less than 3 vertices.

PARAMETERS

n is the number of vertices.

***pFirstCoord** is a pointer to array of vertex coordinates: **x1,y1, x2,y2, x3,y3,**...

RETURN VALUE

None.

SEE ALSO

glPlotPolygon, glFillPolygon, glFillVPolygon

void glPlotPolygon(int n, int y1, int x2, int y2, ...);

Plots the outline of a polygon in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the polygon that is outside the LCD display area will be clipped. The function will also return, doing nothing, if there are less than 3 vertices.

PARAMETERS

n is the number of vertices.

y1 is the *y* coordinate of the first vertex.

x1 is the *x* coordinate of the first vertex.

y2 is the *y* coordinate of the second vertex.

x2 is the *x* coordinate of the second vertex.

... are the coordinates of additional vertices.

RETURN VALUE

None.

SEE ALSO

glPlotVPolygon, glFillPolygon, glFillVPolygon

void glFillVPolygon(int n, int *pFirstCoord);

Fills a polygon in the LCD page buffer and on the LCD screen if the buffer is unlocked. Any portion of the polygon that is outside the LCD display area will be clipped. The function will also return, doing nothing, if there are less than 3 vertices.

PARAMETERS

n is the number of vertices.

***pFirstCoord** is a pointer to array of vertex coordinates: **x1,y1, x2,y2, x3,y3,...**.

RETURN VALUE

None.

SEE ALSO

glFillPolygon, glPlotPolygon, glPlotVPolygon

void glFillPolygon(int n, int x1, int y1, int x2, int y2, ...);

Fills a polygon in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the polygon that is outside the LCD display area will be clipped.

PARAMETERS

n is the number of vertices.

x1 is the *x* coordinate of the first vertex.

y1 is the *y* coordinate of the first vertex.

x2 is the *x* coordinate of the second vertex.

y2 is the *y* coordinate of the second vertex.

... are the coordinates of additional vertices.

RETURN VALUE

None.

SEE ALSO

glFillVPolygon, glPlotPolygon, glPlotVPolygon

void glPlotCircle(int xc, int yc, int rad);

Draws a circle in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the circle that is outside the LCD display area will be clipped.

PARAMETERS

xc is the *x* coordinate of the center of the circle.

yc is the *y* coordinate of the center of the circle.

rad is the radius of the center of the circle (in pixels).

RETURN VALUE

None.

SEE ALSO

glFillCircle, glPlotPolygon, glFillPolygon

void glFillCircle(int xc, int yc, int rad);

Draws a filled circle in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the circle that is outside the LCD display area will be clipped.

PARAMETERS

xc is the *x* coordinate of the center of the circle.

yc is the *y* coordinate of the center of the circle.

rad is the radius of the center of the circle (in pixels).

RETURN VALUE

None.

SEE ALSO

glPlotCircle, glPlotPolygon, glFillPolygon

void glXFontInit(fontInfo *pInfo, char pixWidth, char pixHeight, unsigned startChar, unsigned endChar, unsigned long xmemBuffer);

Initializes the font descriptor structure, where the font is stored in **xmem**. Each font character's bitmap is column major and byte-aligned.

PARAMETERS

***pInfo** is a pointer to the font descriptor to be initialized.

pixWidth is the width (in pixels) of each font item.

pixHeight is the height (in pixels) of each font item.

startChar is the value of the first printable character in the font character set.

endChar is the value of the last printable character in the font character set.

xmemBuffer is the **xmem** pointer to a linear array of font bitmaps.

RETURN VALUE

None.

SEE ALSO

glPrinf

unsigned long glFontCharAddr(fontInfo *pInfo, char letter);

Returns the **xmem** address of the character from the specified font set.

PARAMETERS

***pInfo** is the **xmem** address of the bitmap font set.

letter is an ASCII character.

RETURN VALUE

xmem address of bitmap character font, column major, and byte-aligned.

SEE ALSO

glPutFont, glPrintf

void glPutFont(int x, int y, fontInfo *pInfo, char code);

Puts an entry from the font table to the page buffer and on the LCD if the buffer is unlocked. Each font character's bitmap is column major and byte-aligned. Any portion of the bitmap character that is outside the LCD display area will be clipped.

PARAMETERS

x is the *x* coordinate (column) of the upper left corner of the text.

y is the *y* coordinate (row) of the left top corner of the text.

***pInfo** is a pointer to the font descriptor.

code is the ASCII character to display.

RETURN VALUE

None.

SEE ALSO

glFontCharAddr, glPrintf

void glSetPfStep(int stepX, int stepY);

Sets the **glPrintf()** printing step direction. The *x* and *y* step directions are independent signed values. The actual step increments depend on the height and width of the font being displayed, which are multiplied by the step values.

PARAMETERS

stepX is the **glPrintf** *x* step value

stepY is the **glPrintf** *y* step value

RETURN VALUE

None.

SEE ALSO

Use **glGetPfStep()** to examine the current *x* and *y* printing step direction.

int glGetPfStep(void);

Gets the current **glPrintf()** printing step direction. Each step direction is independent of the other, and is treated as an 8-bit signed value. The actual step increments depends on the height and width of the font being displayed, which are multiplied by the step values.

RETURN VALUE

The *x* step is returned in the MSB, and the *y* step is returned in the LSB of the integer result.

SEE ALSO

Use **glGetPfStep()** to control the *x* and *y* printing step direction.

void glPutChar(char ch, char *ptr, int *cnt, glPutCharInst *pInst)

Provides an interface between the **STDIO** string-handling functions and the graphic library. The **STDIO** string-formatting function will call this function, one character at a time, until the entire formatted string has been parsed. Any portion of the bitmap character that is outside the LCD display area will be clipped.

PARAMETERS

ch is the character to be displayed on the LCD.

***ptr** is not used, but is a place holder for **STDIO** string functions.

***cnt** is not used, is a place holder for **STDIO** string functions.

***pInst** is a font descriptor pointer.

RETURN VALUE

None.

SEE ALSO

glPrintf, glPutFont, doprnt

void glPrintf(int x, int y, fontInfo *pInfo, char *fmt, ...);

Prints a formatted string (much like **printf**) on the LCD screen. Only the character codes that exist in the font set are printed, all others are skipped. For example, '\b', '\t', '\n' and '\r' (ASCII backspace, tab, new line, and carriage return, respectively) will be printed if they exist in the font set, but will not have any effect as control characters. Any portion of the bitmap character that is outside the LCD display area will be clipped.

PARAMETERS

x is the *x* coordinate (column) of the upper left corner of the text.

y is the *y* coordinate (row) of the upper left corner of the text.

***pInfo** is a font descriptor pointer.

***fmt** is a formatted string.

... are formatted string conversion parameter(s).

EXAMPLE

```
glprintf(0,0, &fi12x16, "Test %d\n", count);
```

RETURN VALUE

None.

SEE ALSO

glXFontInit

void glBuffLock(void);

Increments LCD screen locking counter. Graphic calls are recorded in the LCD memory buffer and are not transferred to the LCD if the counter is non-zero.

> **NOTE:** **`glBuffLock()`** and **`glBuffUnlock()`** can be nested up to a level of 255, but be sure to balance the calls. It is not a requirement to use these procedures, but a set of **`glBuffLock()`** and **`glBuffUnlock()`** bracketing a set of related graphic calls speeds up the rendering significantly.

RETURN VALUE

None.

SEE ALSO

`glBuffUnlock, glSwap`

void glBuffUnlock(void);

Decrements the LCD screen locking counter. The contents of the LCD buffer are transferred to the LCD if the counter goes to zero.

RETURN VALUE

None.

SEE ALSO

`glBuffLock, glSwap`

void glSwap(void);

Checks the LCD screen locking counter. The contents of the LCD buffer are transferred to the LCD if the counter is zero.

RETURN VALUE

None.

SEE ALSO

`glBuffUnlock, glBuffLock, _glSwapData` (located in the library specifically for the LCD that you are using)

void glSetBrushType(int type);

Sets the drawing method (or color) of pixels drawn by subsequent graphic calls.

PARAMETER

`type` value can be one of the following macros.

`PIXBLACK` draws black pixels.
`PIXWHITE` draws white pixels.
`PIXXOR` draws old pixel XOR'ed with the new pixel.

RETURN VALUE

None.

SEE ALSO

`glGetBrushType`

int glGetBrushType(void);

Gets the current method (or color) of pixels drawn by subsequent graphic calls.

RETURN VALUE

The current brush type.

SEE ALSO

glSetBrushType

void glPlotDot(int x, int y);

Draws a single pixel in the LCD buffer, and on the LCD if the buffer is unlocked. If the coordinates are outside the LCD display area, the dot will not be plotted.

PARAMETERS

x is the *x* coordinate of the dot.

y is the *y* coordinate of the dot.

RETURN VALUE

None.

SEE ALSO

glPlotline, glPlotPolygon, glPlotCircle

void glPlotLine(int x0, int y0, int x1, int y1);

Draws a line in the LCD buffer, and on the LCD if the buffer is unlocked. Any portion of the line that is beyond the LCD display area will be clipped.

PARAMETERS

x0 is the *x* coordinate of one endpoint of the line.

y0 is the *y* coordinate of one endpoint of the line.

x1 is the *x* coordinate of the other endpoint of the line.

y1 is the *y* coordinate of the other endpoint of the line.

RETURN VALUE

None.

SEE ALSO

glPlotDot, glPlotPolygon, glPlotCircle

void glLeft1(int left, int top, int cols, int rows);

Scrolls byte-aligned window left one pixel, right column is filled by current pixel type (color).

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

RETURN VALUE

None.

SEE ALSO

glHScroll, glRight1

void glRight1(int left, int top, int cols, int rows);

Scrolls byte-aligned window right one pixel, left column is filled by current pixel type (color).

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

RETURN VALUE

None.

SEE ALSO

glHScroll, glLeft1

void glUp1(int left, int top, int cols, int rows);

Scrolls byte-aligned window up one pixel, bottom column is filled by current pixel type (color).

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

RETURN VALUE

None.

SEE ALSO

glVScroll, glDown1

void glDown1(int left, int top, int cols, int rows);

Scrolls byte-aligned window down one pixel, top column is filled by current pixel type (color).

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

RETURN VALUE

None.

SEE ALSO

glVScroll, glUp1

void glHScroll(int left, int top, int cols, int rows, int nPix);

Scrolls right or left, within the defined window by *x* number of pixels. The opposite edge of the scrolled window will be filled in with white pixels. The window must be byte-aligned.

Parameters will be verified for the following:

1. The **left** and **cols** parameters will be verified that they are evenly divisible by 8. If not, they will be changed to a value that is a multiple of 8.
2. Parameters will be checked to verify that the scrolling area is valid. The minimum scrolling area is a width of 8 pixels and a height of one row.

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

nPix is the number of pixels to scroll within the defined window (a negative value will produce a scroll to the left).

RETURN VALUE

None.

SEE ALSO

glVScroll

void glVScroll(int left, int top, int cols, int rows, int nPix);

Scrolls up or down, within the defined window by *x* number of pixels. The opposite edge of the scrolled window will be filled in with white pixels. The window must be byte-aligned.

Parameters will be verified for the following:

1. The **left** and **cols** parameters will be verified that they are evenly divisible by 8. If not, they will be changed to a value that is a multiple of 8.
2. Parameters will be checked to verify that the scrolling area is valid. The minimum scrolling area is a width of 8 pixels and a height of one row.

PARAMETERS

left is the upper left corner of bitmap, must be evenly divisible by 8.

top is the left top corner of the bitmap.

cols is the number of columns in the window, must be evenly divisible by 8.

rows is the number of rows in the window.

nPix is the number of pixels to scroll within the defined window (a negative value will produce a scroll up).

RETURN VALUE

None.

SEE ALSO

glHScroll

void glXPutBitmap(int left, int top, int width, int height, unsigned long bitmap);

Draws bitmap in the specified space. The data for the bitmap are stored in **xmem**. This function calls **glXPutFastmap** automatically if the bitmap is byte-aligned (the left edge and the width are each evenly divisible by 8).

Any portion of a bitmap image or character that is outside the LCD display area will be clipped.

PARAMETERS

left is the upper left corner of the bitmap.

top is the upper left corner of the bitmap.

width is the width of the bitmap.

height is the height of the bitmap.

bitmap is the address of the bitmap in **xmem**.

RETURN VALUE

None.

SEE ALSO

glXPutFastmap, glPrintf

void glXPutFastmap(int left, int top, int width, int height, unsigned long bitmap);

Draws bitmap in the specified space. The data for the bitmap are stored in **xmem**. This function is like **glXPutBitmap**, except that it is faster. The restriction is that the bitmap must be byte-aligned.

Any portion of a bitmap image or character that is outside the LCD display area will be clipped.

PARAMETERS

left is the upper left corner of the bitmap, must be evenly divisible by 8.

top is the upper left corner of the bitmap.

width is the width of the bitmap, must be evenly divisible by 8.

height is the height of the bitmap.

bitmap is the address of the bitmap in **xmem**.

RETURN VALUE

None.

SEE ALSO

glXPutBitmap, glPrintf

int TextWindowFrame(windowFrame *window, fontInfo *pFont, int x, int y, int winWidth, int winHeight)

Defines a text-only display window. This function provides a way to display characters within the text window using only character row and column coordinates. The text window feature provides end-of-line wrapping and clipping after the character in the last column and row is displayed.

NOTE: Execute the **TextWindowFrame** function before other **Text...** functions.

PARAMETERS

***window** is a window frame descriptor pointer.

***pFont** is a font descriptor pointer.

x is the *x* coordinate of where the text window frame is to start.

y is the *y* coordinate of where the text window frame is to start.

winWidth is the width of the text window frame.

winHeight is the height of the text window frame.

RETURN VALUE

0—window frame was successfully created.
-1—*x* coordinate + width has exceeded the display boundary.
-2—*y* coordinate + height has exceeded the display boundary.

void TextGotoXY(windowFrame *window, int col, int row);

Sets the cursor location on the display of where to display the next character. The display location is based on the height and width of the character to be displayed.

NOTE: Execute the **TextWindowFrame** function before using this function.

PARAMETERS

***window** is a pointer to a font descriptor.

col is a character column location.

row is a character row location.

RETURN VALUE

None.

SEE ALSO

TextPutChar, TextPrintf, TextWindowFrame

void TextCursorLocation(windowFrame *window, int *col, int *row);

Gets the current cursor location that was set by a Graphic **Text...** function.

NOTE: Execute the **TextWindowFrame** function before using this function.

PARAMETERS

***window** is a pointer to a font descriptor.

***col** is a pointer to cursor column variable.

***row** is a pointer to cursor row variable.

RETURN VALUE

Lower word = Cursor Row location
Upper word = Cursor Column location

SEE ALSO

TextGotoXY, TextPrintf, TextWindowFrame, TextCursorLocation

void TextPutChar(struct windowFrame *window, char ch);

Displays a character on the display where the cursor is currently pointing. If any portion of a bitmap character is outside the LCD display area, the character will not be displayed.

NOTE: Execute the **TextWindowFrame** function before using this function.

PARAMETERS

***window** is a pointer to a font descriptor.

ch is a character to be displayed on the LCD.

RETURN VALUE

None.

SEE ALSO

TextGotoXY, TextPrintf, TextWindowFrame, TextCursorLocation

void TextPrintf(struct windowFrame *window, char *fmt, ...);

Prints a formatted string (much like **printf**) on the LCD screen. Only printable characters in the font set are printed, also escape sequences, '\r' and '\n' are recognized. All other escape sequences will be skipped over; for example, '\b' and '\t' will print if they exist in the font set, but will not have any effect as control characters.

The text window feature provides end-of-line wrapping and clipping after the character in the last column and row is displayed.

NOTE: Execute the **TextWindowFrame** function before using this function.

PARAMETERS

***window** is a pointer to a font descriptor.

***fmt** is a formatted string.

... are formatted string conversion parameter(s).

EXAMPLE

```
TextPrintf(&TextWindow, "Test %d\n", count);
```

RETURN VALUE

None.

SEE ALSO

TextGotoXY, TextPutChar, TextWindowFrame, TextCursorLocation

B.7.3 Keypad

The functions used to control the keypad are contained in the **KEYPAD7.LIB** library located in the Dynamic C **KEYPADS** library directory.

```
void keyInit(void);
```

Initializes keypad process

RETURN VALUE

None.

SEE ALSO

brdInit

```
void keyConfig(char cRaw, char cPress,
   char cRelease, char cCntHold, char cSpdLo,
   char cCntLo, char cSpdHi);
```

Assigns each key with key press and release codes, and hold and repeat ticks for auto repeat and debouncing.

PARAMETERS

cRaw is a raw key code index.

1x7 keypad matrix with raw key code index assignments (in brackets):

[0]		[1]		[2]		[3]
	[4]		[5]		[6]	

User Keypad Interface

cPress is a key press code

An 8-bit value is returned when a key is pressed.
0 = Unused.

See **keypadDef()** for default press codes.

cRelease is a key release code.

An 8-bit value is returned when a key is pressed.
0 = Unused.

cCntHold is a hold tick.

How long to hold before repeating.
0 = No Repeat.

cSpdLo is a low-speed repeat tick.

How many times to repeat.
0 = None.

cCntLo is a low-speed hold tick.

How long to hold before going to high-speed repeat.
0 = Slow Only.

cSpdHi is a high-speed repeat tick.

How many times to repeat after low speed repeat.
0 = None.

RETURN VALUE

None.

SEE ALSO

keyProcess, keyGet, keypadDef

void keyProcess(void);

Scans and processes keypad data for key assignment, debouncing, press and release, and repeat.

NOTE: This function is also able to process an 8 × 8 matrix keypad.

RETURN VALUE

None

SEE ALSO

keyConfig, keyGet, keypadDef

char keyGet(void);

Get next keypress

RETURN VALUE

The next keypress, or 0 if none

SEE ALSO

keyConfig, keyProcess, keypadDef

int keyUnget(char cKey);

Push keypress on top of input queue

PARAMETER

cKey is the key being pressed

RETURN VALUE

None.

SEE ALSO

keyGet

void keypadDef();

Configures the physical layout of the keypad with the desired ASCII return key codes.

Keypad physical mapping 1 × 7

0	4	1	5	2	6	3
['L']		['U']		['D']		['R']
	['–']		['+']		['E']	

where

'E' represents the ENTER key
'D' represents Down Scroll
'U' represents Up Scroll
'R' represents Right Scroll
'L' represents Left Scroll

Example: Do the followingfor the above physical vs. ASCII return key codes.

```
keyConfig (  3,'R',0, 0, 0, 0, 0 );
keyConfig (  6,'E',0, 0, 0, 0, 0 );
keyConfig (  2,'D',0, 0, 0, 0, 0 );
keyConfig (  4,'-',0, 0, 0, 0, 0 );
keyConfig (  1,'U',0, 0, 0, 0, 0 );
keyConfig (  5,'+',0, 0, 0, 0, 0 );
keyConfig (  0,'L',0, 0, 0, 0, 0 );
```

Characters are returned upon keypress with no repeat.

RETURN VALUE

None.

SEE ALSO

keyConfig, keyGet, keyProcess

void keyScan(char *pcKeys);

Writes "1" to each row and reads the value. The position of a keypress is indicated by a zero value in a bit position.

PARAMETER

***pcKeys** is the address of the value read.

RETURN VALUE

None.

SEE ALSO

keyConfig, keyGet, keypadDef, keyProcess

B.8 Sample Programs

The following sample programs are found in the **122x32_1x7** subdirectory in **SAMPLES\LCD_Keypad**.

- **ALPHANUM.C**—Demonstrates how to create messages using the keypad and then displaying them on the LCD display.
- **COFTERMA.C**—Demonstrates cofunctions, the cofunction serial library, and using a serial ANSI terminal such as Hyperterminal from an available COM port connection.
- **DISPPONG.C**—Demonstrates output to LCD display.
- **DKADEMO1.C**—Demonstrates some of the LCD/keypad module font and bitmap manipulation features with horizontal and vertical scrolling, and using the **GRAPHIC.LIB** library.
- **FUN.C**—Demonstrates drawing primitive features (lines, circles, polygons) using the **GRAPHIC.LIB** library
- **KEYBASIC.C**—Demonstrates the following keypad functions in the **STDIO** display window:
  - default ASCII keypad return values.
  - custom ASCII keypad return values.
  - keypad repeat functionality.
- **KEYMENU.C**—Demonstrates how to implement a menu system using a highlight bar on a graphic LCD display. The menu options for this sample are as follows.
  1. Set Date/Time
  2. Display Date/Time
  3. Turn Backlight OFF
  4. Turn Backlight ON
  5. Toggle LEDs
  6. Increment LEDs
  7. Disable LEDs
- **LED.C**—Demonstrates how to toggle the LEDs on the LCD/keypad module.
- **SCROLLING.C**—Demonstrates scrolling features of the **GRAPHIC.LIB** library.
- **TEXT.C**—Demonstrates the text functions in the **GRAPHIC.LIB** library. Here is a list of what is demonstrated.
  1. Font initialization.
  2. Text window initialization.
  3. Text window, end-of-line wraparound, end-of-text window clipping, line feed, and carriage return.
  4. Creating 2 different TEXT windows for display.
  5. Displaying different FONT sizes.

The following sample programs, found in the **TCPIP** subdirectory in **SAMPLES/LCD_Keypad/122x32_1x7**, are targeted at the Ethernet-enabled versions of the Smart Star, the Smart Star and the BL2110. Remember to configure the IP address, netmask, and gateway as indicated in the sample programs.

- **MBOXDEMO.C**—This program implements a web server that allows e-mail messages to be entered that are then shown on the LCD display. The keypad allows you to scroll within messages, flip to other e-mails, mark messages as read, and delete e-mails. When a new e-mail arrives, an LED turns on, and turns off once the message has been marked as read. A log of all e-mail actions is kept, and can be displayed in the Web browser. All current e-mails can also be read with the Web browser.

  When using **MBOXDEMO.C**, connect the Smart Star and a PC (or other device with a Web Browser) to an Ethernet. If you connect the PC and the Smart Star directly, be sure to use a crossover Ethernet cable; strait-through Ethernet cables and a hub may be used instead.

- **TCP_RESPOND.C**—This program and **TCP_SEND.C** are executed on two separate single-board computers to demonstrate how the two boards communicate with each other. Use **PCSEND.EXE** on the PC console side at the command prompt if you do not have a second board. **PCSEND.EXE** is located with source code in the **SAMPLES/LCD_Keypad/Windows** directory.

  TCP_RESPOND.C waits for a message from another single-board computer. The message received is displayed on the LCD, and you may respond by pressing a key on the keypad. The response is then sent to the remote single-board computer.

- **TCPSEND.C**—This program and **TCP_RESPOND.C** are executed on two separate single-board computers to demonstrate how the two boards communicate with each other. Use **PCRESPOND.EXE** on the PC console side at the command prompt if you do not have a second board. **PCRESPOND.EXE** is located with source code in the **SAMPLES/LCD_Keypad/Windows** directory.

  When a key on the keypad is pressed, a message associated with that key is sent to a specified destination address and port. The destination then responds to that message. The response is displayed on the LCD.

  Note that only the **LEFT** and **UP** scroll keys are set up to cause a message to be sent.

When using **TCPSEND.C** and **TCP_RESPOND.C**, connect the Smart Star and the other single-board computer to an Ethernet. If you connect the them directly, be sure to use a crossover Ethernet cable; straight-through Ethernet cables and a hub may be used instead.

APPENDIX C. POWER MANAGEMENT

Appendix C provides information on the current requirements of the Smart Star I/O cards, the use and installation of a backup battery, and some background on power mangement.

C.1 Current Requirements

Remember to take the current draw of the various I/O cards into consideration when selecting the power supply for your Smart Star control system.

Table C-1 lists the typical current consumption for the CPU card and the I/O cards.

Table C-1. Current Consumption of I/O Cards Attached to Smart Star Backplane

I/O Cards	Current Consumption	
	+5 V Supply	+V_USER Supply
Digital I/O (SR9200 series)	65 mA	up to 200 mA/output*
A/D Converter (SR9300 series) D/A Converter (SR9400 series)	40 mA	35 mA
Relay (SR9500 series)	10 mA	75 mA
CPU card	190 mA	—

* Maximum current 2.0 A per I/O card, 7.0 A for Smart Star system

C.2 Batteries and External Battery Connections

An onboard 265 mA·h lithium coin cell on the CPU card provides power to the real-time clock and SRAM when external power is removed from the Smart Star control system. This allows the CPU card to continue to keep track of time and preserves the SRAM memory contents while the power is off.

The drain on the battery is typically less than 20 µA when there is no external power applied. The battery can last

$$\frac{265\ \text{mA}\cdot\text{h}}{10\ \mu\text{A}} = 3.0\ \text{years.}$$

The drain on the battery is typically less than 4 µA when external power ***is*** applied. The battery can last for

$$\frac{265\ \text{mA}\cdot\text{h}}{4\ \mu\text{A}} = 7.5\ \text{years.}$$

Since the shelf life of the battery is 10 years, the battery can last for most of its shelf life when external power is applied most of the time.

C.2.1 Replacing the Backup Battery

The battery is user-replaceable, and is fitted in a battery holder. To replace the battery, lift up on the spring clip and slide out the old battery. Use only a Panasonic CR2330 or equivalent replacement battery, and insert it into the battery holder with the + side facing up.

NOTE: The SRAM contents and the real-time clock settings will be lost if the battery is replaced with no power applied to the Smart Star. There is a provision for an external battery if you need to save the SRAM contents and the real-time clock settings since the CPU card needs to be removed from the backplane in order to change the onboard battery.

CAUTION: There is an explosion danger if the battery is short-circuited, recharged, or replaced incorrectly. Replace the battery only with the same type or an equivalent type recommended by the battery manufacturer. Dispose of used batteries according to the battery manufacturer's instructions.

C.2.2 Battery-Backup Circuit

Figure C-1 shows the battery-backup circuit.

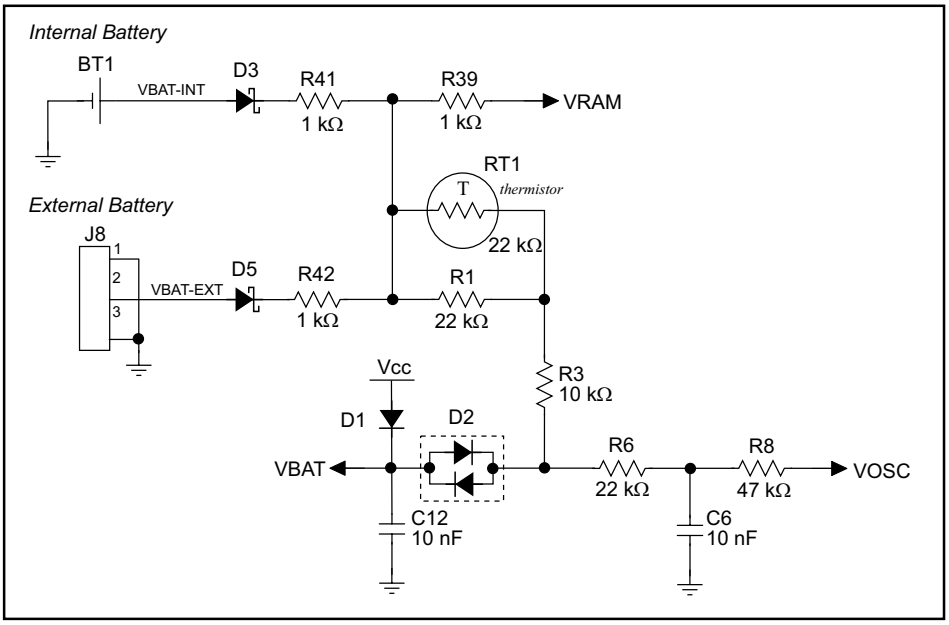


Figure C-1. Smart Star CPU Card Backup Battery Circuit

The battery-backup circuit serves three purposes:

- It reduces the battery voltage to the SRAM and to the real-time clock, thereby limiting the current consumed by the real-time clock and lengthening the battery life.
- It ensures that current can flow only ***out*** of the battery to prevent charging the battery.
- A voltage, VOSC, is supplied to U14, which keeps the 32.768 kHz oscillator working when the voltage begins to drop.

VRAM and Vcc are nearly equal (<100 mV, typically 10 mV) when power is supplied to the CPU card.

C.2.3 Power to VRAM Switch

The VRAM switch, shown in Figure C-2, allows the battery backup to provide power when the external power goes off. The switch provides an isolation between +5 V and the battery when +5 V goes low. This prevents the +5 V line from draining the battery.

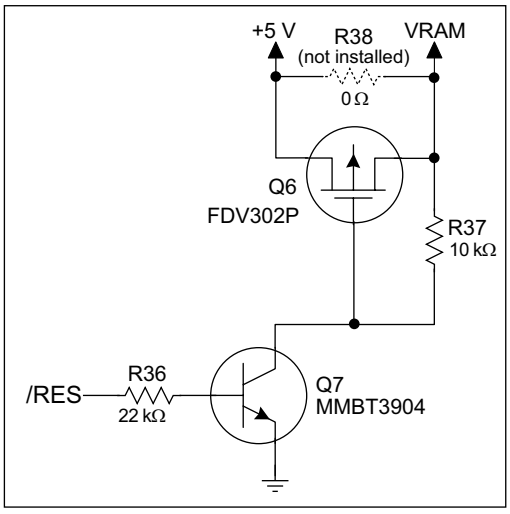


Figure C-2. VRAM Switch

Transistor Q6 is needed to provide a very small voltage drop between +5 V and VRAM (<100 mV, typically 10 mV) so that the processor lines powered by +5 V will not have a significantly different voltage than VRAM.

When the CPU card is ***not*** resetting (pin 2 on U4 is high), the /RES line will be high. This turns on Q6, causing its collector to go low. This turns on Q7, allowing VRAM to nearly equal +5 V.

When the CPU card ***is*** resetting, the /RES line will go low. This turns off Q6 and Q7, providing an isolation between +5 V and VRAM.

The battery-backup circuit keeps VRAM from dropping below 2 V.

C.2.4 Reset Generator

The CPU card uses a reset generator, U4, to reset the Rabbit 2000 microprocessor when the voltage drops below the voltage necessary for reliable operation. The reset occurs between 4.50 V and 4.75 V, typically 4.63 V.

C.2.5 External Battery

A connection for an external backup battery is provided at header J8, shown in Figure C-3. The header is wired to provide reverse polarity protection.

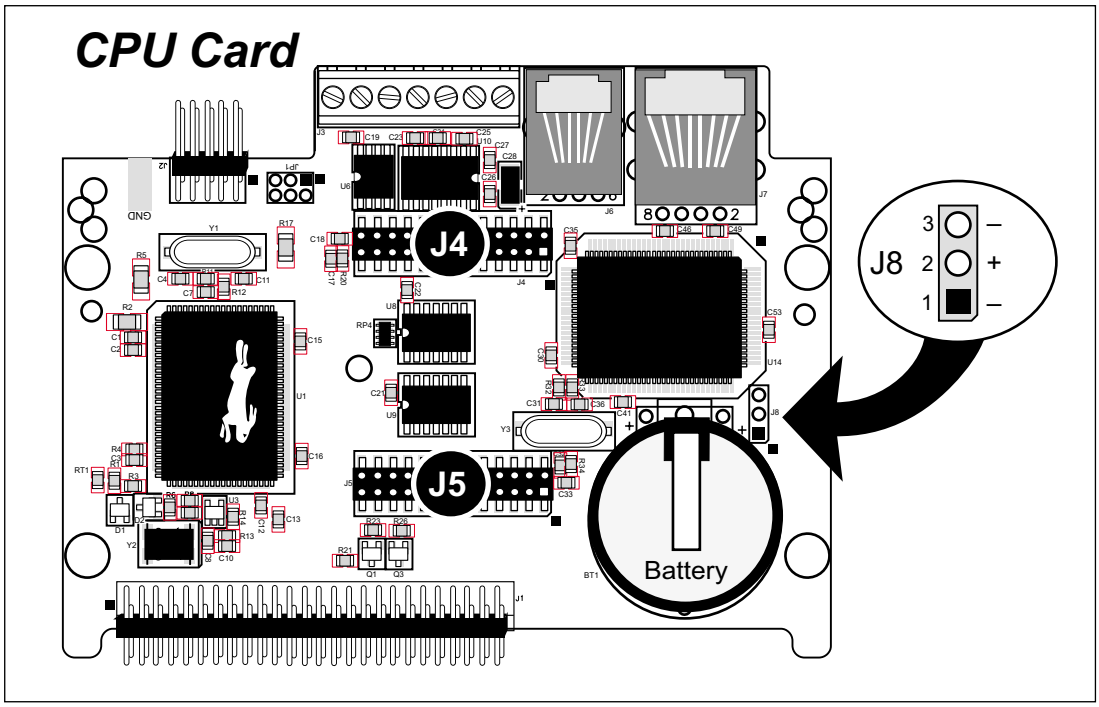


Figure C-3. External Backup Battery Connection

The external battery connection is useful if the SRAM and real-time clock data need to be preserved while the backup battery is being changed. This way power can continue to be applied to the CPU card from the backplane (if the external backup battery is being replaced) or from the external battery (if the onboard backup battery needs to be changed since this requires removing the CPU card from the backplane in order to access the onboard backup battery).

C.3 Chip Select Circuit

Figure C-4 shows a schematic of the chip select circuit for the RAM.

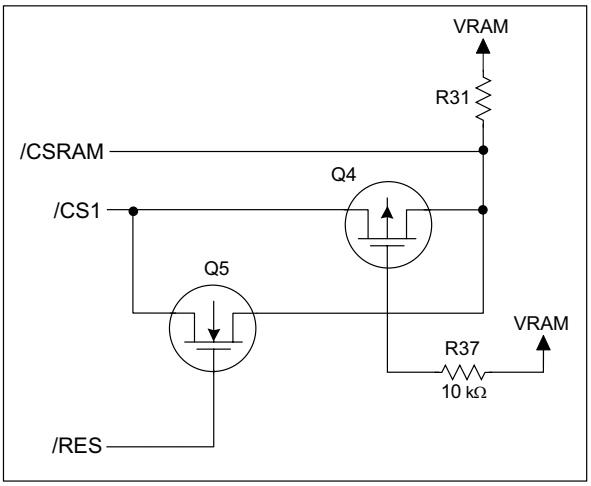


Figure C-4. Chip Select Circuit

The current drain on the battery in a battery-backed circuit must be kept to a minimum. When the CPU card is not powered, the battery keeps the SRAM memory contents and the real-time clock (RTC) going. The SRAM has a powerdown mode that greatly reduces power consumption. This powerdown mode is activated by raising the chip select (CS) signal line. Normally the SRAM requires +5 V to operate. However, only 2 V is required for data retention in powerdown mode. Thus, when power is removed from the circuit, the battery voltage needs to be provided to both the SRAM power pin and to the CS signal line. The CS control circuit accomplishes this task for the CS signal line.

In a powered-up condition, the CS control circuit must allow the processor's chip select signal /CS1 to control the SRAM's CS signal /CSRAM. So, with power applied, /CSRAM must be the same signal as /CS1, and with power removed, /CSRAM must be held high (but only needs to be battery voltage high). Q4 and Q5 are MOSFET transistors with opposing polarity. They are both turned on when power is applied to the circuit. They allow the CS signal to pass from the processor to the SRAM so that the processor can periodically access the SRAM. When power is removed from the circuit, the transistors will turn off and isolate /CSRAM from the processor. The isolated /CSRAM line has a 100 kΩ pullup resistor to VRAM (R31). This pullup resistor keeps /CSRAM at the VRAM voltage level (which under no power condition is the backup battery's regulated voltage at a little more than 2 V).

Transistors Q4 and Q5 are of opposite polarity so that a rail-to-rail voltage can be passed. When the /CS1 voltage is low, Q5 will conduct. When the /CS1 voltage is high, Q4 will conduct. It takes time for the transistors to turn on, creating a propagation delay. This delay is typically very small, about 10 ns to 15 ns.

The signal that turns the transistors on is a high on the processor's reset line, /RES. When the CPU card is not in reset, the reset line will be high, turning on n-channel Q5. When a reset occurs, the /RES line will go low.

APPENDIX D. PROGRAMMING CABLE

Appendix D provides additional theoretical information for the Rabbit 2000™ microprocessor when using the **DIAG** and **PROG** connectors on the programming cable with the CPU card in the Smart Star system. The **PROG** connector is used only when the programming cable is attached to the proramming connector (header J2) on the CPU card while a new application is being developed. Otherwise, the **DIAG** connector on the programming cable allows the programming cable to be used as an RS-232 to CMOS level converter for serial communication, which is appropriate for monitoring or debugging the Smart Star system while it is running.

The programming port, which is shown in Figure D-1, can serve as a convenient communications port for field setup or other occasional communication need (for example, as a diagnostic port). There are several ways that the port can be automatically integrated into software. If the port is simply to perform a setup function, that is, write setup information to flash memory, then the controller can be reset through the programming port and a cold boot performed to start execution of a special program dedicated to this functionality.

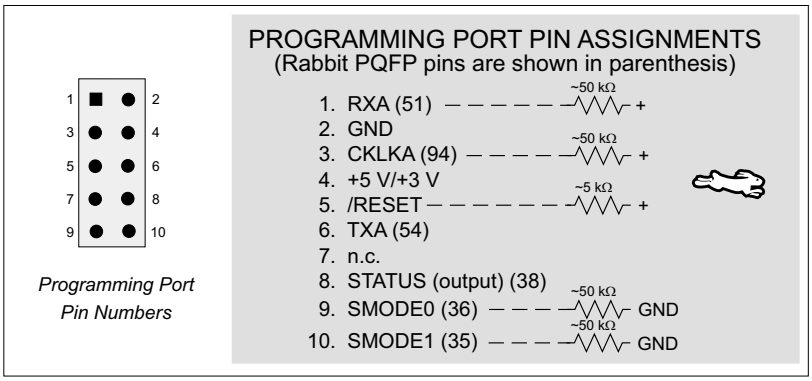


Figure D-1. Programming Port Pin Assignments

When the **PROG** connector is used, the /RESET line can be asserted by manipulating DTR and the STATUS line can be read as DSR on the serial port. The target can be restarted by pulsing reset and then, after a short delay, sending a special character string at 2400 bps. To simply restart the BIOS, the string 80h, 24h, 80h can be sent. When the BIOS is started, it can tell whether the programming cable is connected because the SMODE1 and SMODE0 pins are sensed as being high. This will cause the Rabbit 2000 to enter the bootstrap mode. The Dynamic C programming mode then can have an escape message that will enable the diagnostic serial port function.

Alternatively, the **DIAG** connector can be used to connect the programming port. The /RESET line and the SMODE1 and SMODE0 pins are not connected to this connector. The programming port is then enabled as a diagnostic port by polling the port periodically to see if communication needs to begin or to enable the port and wait for interrupts. The pull-up resistors on RXA and CLKA prevent spurious data reception that might take place if the pins floated.

If the clocked serial mode is used, the serial port can be driven by having two toggling lines that can be driven and one line that can be sensed. This allows a conversation with a device that does not have an asynchronous serial port but that has two output signal lines and one input signal line.

The line TXA (also called PC6) is zero after reset if the cold-boot mode is not enabled. A possible way to detect the presence of a cable on the programming port is for the cable to connect TXA to one of the SMODE pins and then test for the connection by raising PC6 and reading the SMODE pin after the cold-boot mode has been disabled. The value of the SMODE pin is read from the SPCR register.

APPENDIX E.
SMART STAR SLOT ADDRESS LAYOUT

Appendix F provides information about the register addresses for the various I/O card slots on the backplane. The information in this appendix will be of interest to more advanced users.

The slots on the Smart Star backplane are accessed as external registers via the Rabbit 2000's assembly **IOE** prefix or via standard Rabbit BIOS functions. More convenient functions specific to the Smart Star control system have been written to provide more flexibility; for example, there is now a provision for the automatic update of shadow registers for each slot and for each register.

The Smart Star design routes four address bits to each slot, providing 16 register addresses for each slot. These bits are passed through as bits 0–3 of the register address. The slot number itself is assigned to bits 6–8 of the address. In addition, the backplane design requires that bits 13 and 14 be high and that bit 9 be low. The simplest way to enforce this is to use a base address of 0x6000. Table E-1 provides the address layout for accessing the Smart Star slots, where S*n* is the binary representation of the slot number (0–6), R*n* is the binary representation of the register numbers (0–15), and *X* means the value does not matter.

Table E-1. Smart Star External Register Address Bitmap

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	1	1	0	*X*	*X*	0	S2	S1	S0	*X*	*X*	R3	R2	R1	R0

This bit mapping of the external register address provides the register addresses for each slot as listed in Table E-2.

Table E-2. Slot External Register Addresses

Slot Number	Address Range
0	0x6000–0x600F
1	0x6040–0x604F
2	0x6080–0x608F
3	0x60C0–0x60CF
4	0x6100–0x610F
5	0x6140–0x614F
6	0x6180–0x618F

E.1 Digital I/O Card Channel Layout

The digital I/O card layout is complicated by the standard Z-World method of minimizing chip layout while adding channel arrangement flexibility. In particular, the nibble-wise layout of digital input channels requires fewer chips if fewer channels are desired. This is a common feature on Z-World products and should not surprise most users. The digital output channel layout is straightforward.

It is also possible to access the digital I/O channels in banks of eight channels. This method is significantly faster than reading eight channels one at a time, and so was included in the API.

Table E-3. Digital I/O Card Bank/Channel Mapping

Local Board Address	Input Bank	Output Bank	Input Channels	Output Channels
0x00	0		0–3/8–11	
0x01	2		4–7/12–15	
0x02		1		0–7
0x03		2		8–15

E.2 A/D Converter Card Channel Layout

The A/D converter card contains a single 11-input 12-bit A/D converter, TLC2543. The method of interfacing to this chip is a combination of single-bit writes via board registers and synchronous clocked serial access via the CPU card's Serial Port B, which is extended across all eight slots. In addition, a serial EEPROM is installed on the A/D converter card to store the calibration constants.

Table E-4. A/D Converter Card Control Registers

Address	Data Bits	Value	Description
0x0	Write D7–D0	D7–D4 selects input channel, D3–D0 selects conversion channel	Load A/D converter with data byte
0x0	Read D1	0	A/D converter end of conversion signal
		1	A/D converter busy
0x1	Write D0	0	Enable A/D conversion
		1	Disable A/D conversion
0x2	Write D0	0	EEPROM clock line low
		1	EEPROM clock line high
0x3	Write D0	0	EEPROM data line low
		1	EEPROM data line high
0x0	Read D2	0	EEPROM acknowledge signal
		1	EEPROM busy

E.3 D/A Converter Card Channel Layout

The D/A converter card contains four two-channel 12-bit D/A converters, TLV5618, to produce 8 analog output channels. Each channel is accessed by the slot, channel and device addressing scheme. The D/A converter card also has an EEPROM to store calibration constants.

Table E-5. D/A Converter Card Control Registers

Address	Data Bits	Value	Description
0x0	D0	0	D/A converter clock line low
		1	D/A converter clock line high
	D1	X	D/A converter data input line
	D2	0	D/A converter chip select channels 0 and 1
	D3	0	D/A converter chip select channels 2 and 3
	D4	0	D/A converter chip select channels 4 and 5
	D5	0	D/A converter chip select channels 6 and 7
	D6	0	EEPROM clock line low
		1	EEPROM clock line high
	D7	X	EEPROM data line

External reads and writes (/IORD and /IOWR) control the data direction.

E.4 Relay Card Channel Layout

The relay card layout is complemented by the standard Z-World method of minimizing chip layout while adding channel arrangement flexibility. In particular, the nibble-wise layout of the relay channels requires fewer chips if fewer channels are desired. This is a common feature on Z-World products and should not surprise most users. The relay channel layout is straightforward.

Table E-6. Relay Card Channel Mapping

Local Board Address	SR9500 Relay Channels	SR9510 Relay Channels
0x00	REL0	REL0
0x01	REL1	REL1
0x02	REL2	REL2
0x03	REL3	REL3
0x04	REL4	REL4
0x05	REL5	REL5
0x06	—	REL6
0x07	—	REL7

INDEX

J

K

L

M

O

P

R

S

T

SCHEMATICS

090-0129 CPU Card (SR9150) Schematic

090-0143 Backplane (SR9010) Schematic

090-0130 Backplane (SR9050) Schematic

090-0101 Digital I/O Card–Sinking (SR9200) Schematic

090-0118 Digital I/O Card–Sourcing (SR92x5) Schematic

090-0086 A/D Converter Card (SR9300) Schematic

090-0121 D/A Converter Card (SR9400) Schematic

090-0098 6-Relay Card (SR9500) Schematic

090-0108 8-Relay Card (SR9510) Schematic

090-0102 FWT18 Schematic

090-0106 FWT18R Schematic

090-0103 FWT27 Schematic

090-0125 LCD/Keypad Module Schematic

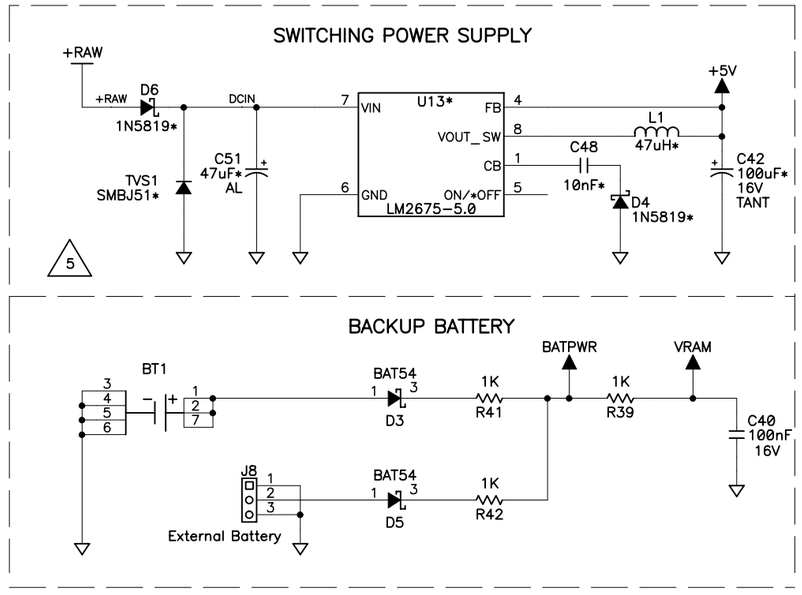
090-0128 Programming Cable Schematic

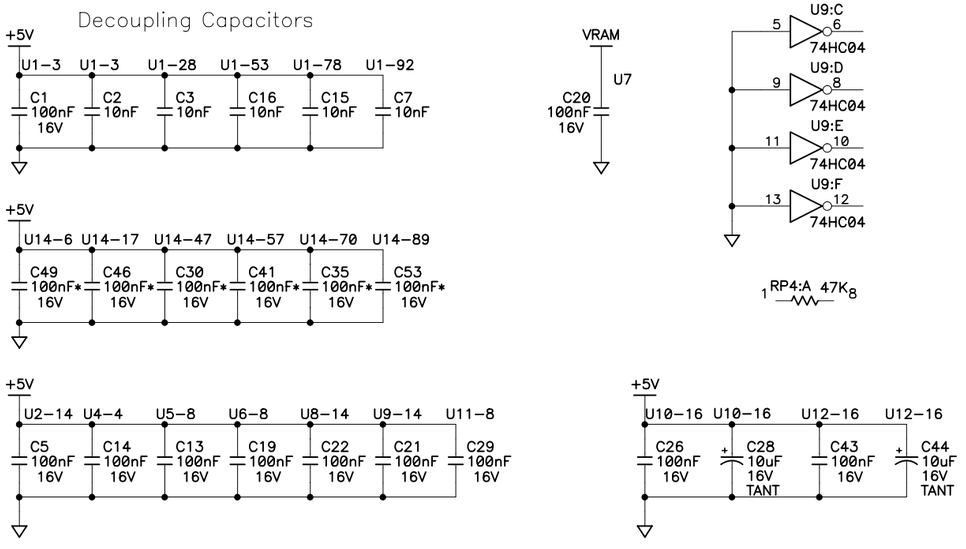
NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/16W, 5%
2. ALL JUMPER RESISTORS ARE 0 OHMS 0805 PACKAGE.
3. ALL CAPACITORS ARE 50VDC OR HIGHER.
4. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (Vcc), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

5 OUTLINED CIRCUIT NOT STUFFED.

6. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.





REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11809	INITIAL RELEASE	XT	2/4/02	KIS	2/4/02
B	E11834	REMOVE UNUSED REFERENCES FOR CLARIFICATION	XT	2/15/02	KlS	2/15/02

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION				DEVICE: FILTER CAP REF DES(s)
		GND	+5V	VRAM	NO CONNECTS	
U1	RABBIT 2000	2,27,39, 52,77,89	3,28,53,78,92			PIN3 – C14 + C15 C23, C24, C17, C16
U2	74HC08	7	14			C5
U3	74AHC1GU04	3	5			
U4	ETC811L	1	4			C14
U5	FLASH	24	8			C13
U6	SP483E	5	8			C19
U7	SRAM 512K X 8	16		32		C20
	SRAM 128K X 8	16		32		
U8	74HC125	7	14			C22
U9	74HC04	7	14			C21
U10	232A	15	16			C26, C28
U11	FLASH	24	8			C29
U12	232A	15	16			C43, C44
U13	LM2575-5.0				2-6,9,11,13,16, 19-24	
U14	RTL8019AS	14,28,44, 52,83,86	6,17,47, 57,70,89			C48, C44, C32, C45, C46, C49

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) XUAN TRUONG	2/4/02
REVISED BY: KRISTI SCHALLER	2/12/02
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: XUAN TRUONG	2/4/02
ENGINEERING MANAGER: R.MATTHEWS	2/4/02
SIGNATURES	DATE

TITLE

SCHEMATIC DIAGRAM SR9150/9160 SMART STAR CPU

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE B | DWG NO. 090-0129

SCALE NONE | RELEASE DATE 2/4/02 | SHEET 1 OF 5

U14: OPTIONAL 512K OR 128K

<3,4>D0–D7
<3,4>A0–11
<3,4> SLOT–P
<3>RS232/RS485
<4>PROGRAM_OUT
<3,4>ETHERNET

D0–7
A0–11
SLOT–P
RS232/RS485

U1
RABBIT 2000

SRAM SELECT*
(0 ohm) Resistor Jumper
128K SRAM 1–2
512K SRAM 2–3

JP3 VRAM
SRAM_SEL
U7 SRAM
SRAM TYPE: 512K X 8 | 128K X 8 | PIN ID A17 CS2 | A18 NC

ADDR. BANK SELECT*
(0 ohm) Resistor Jumper
NORMAL 1–2

JP5
F1_A17
F1_A18
U11 FLASH
U5 FLASH
FLASH TYPE: 512K X 8 | 256K X 8 | 128K X 8 | PIN ID A17 A17 NC | A18 NC/+RST NC

Reset Generator
U4 ETC811L
C9 10nF
R10 100K
R16 10K
U2:B 74HC08
U2:A 74HC08
R9 10K

FLASH 2 SELECT
JP2
1–2 128K/256K FLASH
2–3 512K FLASH
F2_A18

FLASH 1 SELECT
JP4
1–2 128K/256K FLASH
2–3 512K FLASH
F1_A18

CS Control
R38(0 ohm*) R35 (0 ohm*) R31 100K
R37 10K
R36 22K
R15 47K
Q6 FDV302P
Q4 FDV302P
Q7 MMBT3904
Q5 2N7002
TP2 TP3

R21 1K /ATTN1
R7 1K /ATTN2
PE2
SLOT_ENABLE
CPU_PE6–/SCS
INHIB_OUT

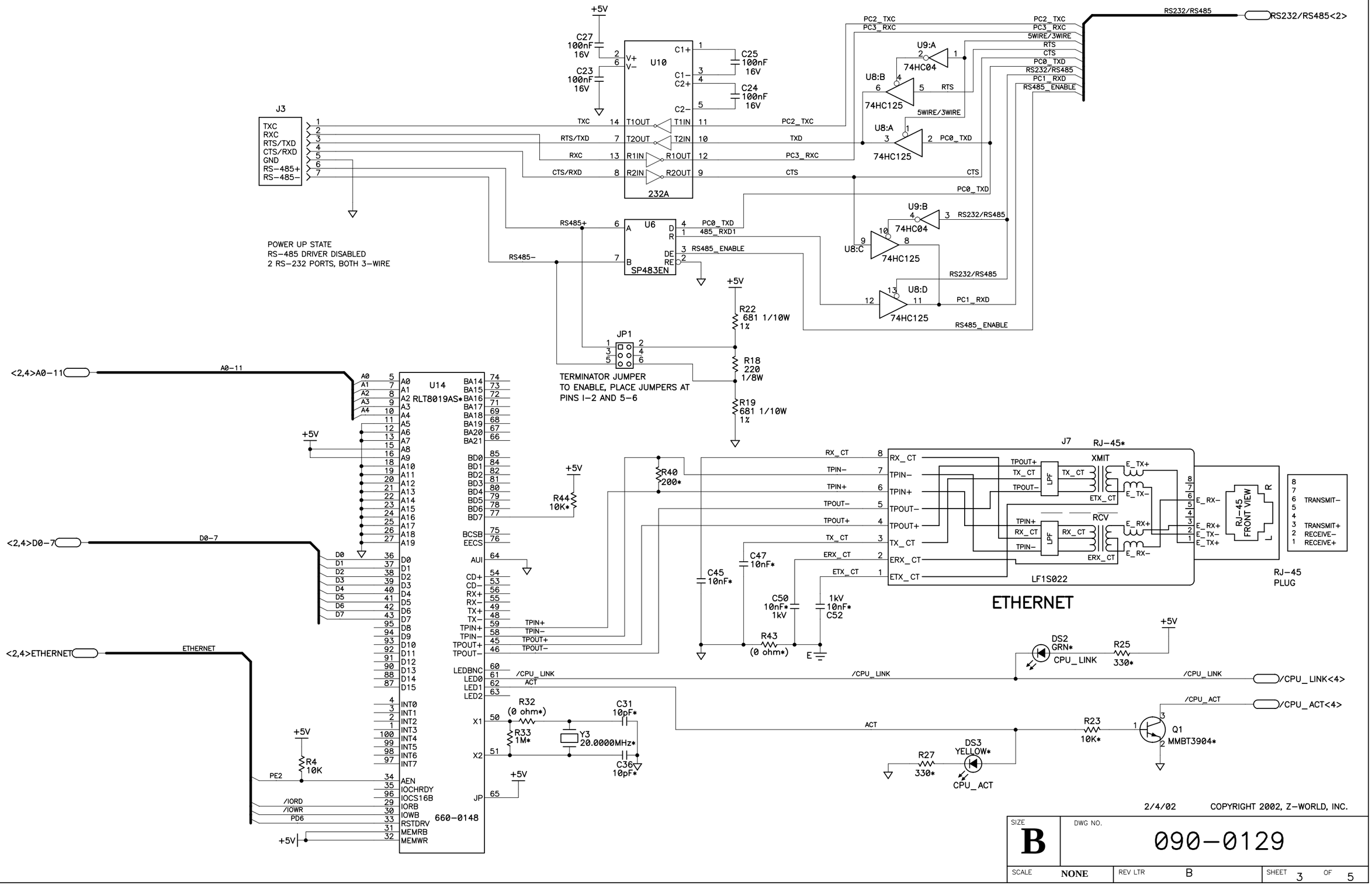
RN2 47k PA0–PA7
RN1 47k PB0–PB7
RP3:D RP3:C RP3:B RP3:A 47K CPU_RXA RXB PC3_RXC PC1_RXD
RP2:D RP2:C RP2:B RP2:A 47K RS232/RS485 PD5 PD6 CPU_PE6–/SCS
RP1:A RP1:B 47K /ATTN1 /ATTN2
RP4:D RP4:C RP4:B 47K RTS 5WIRE/3WIRE RS485_ENABLE
RP1:C RP1:D 47K

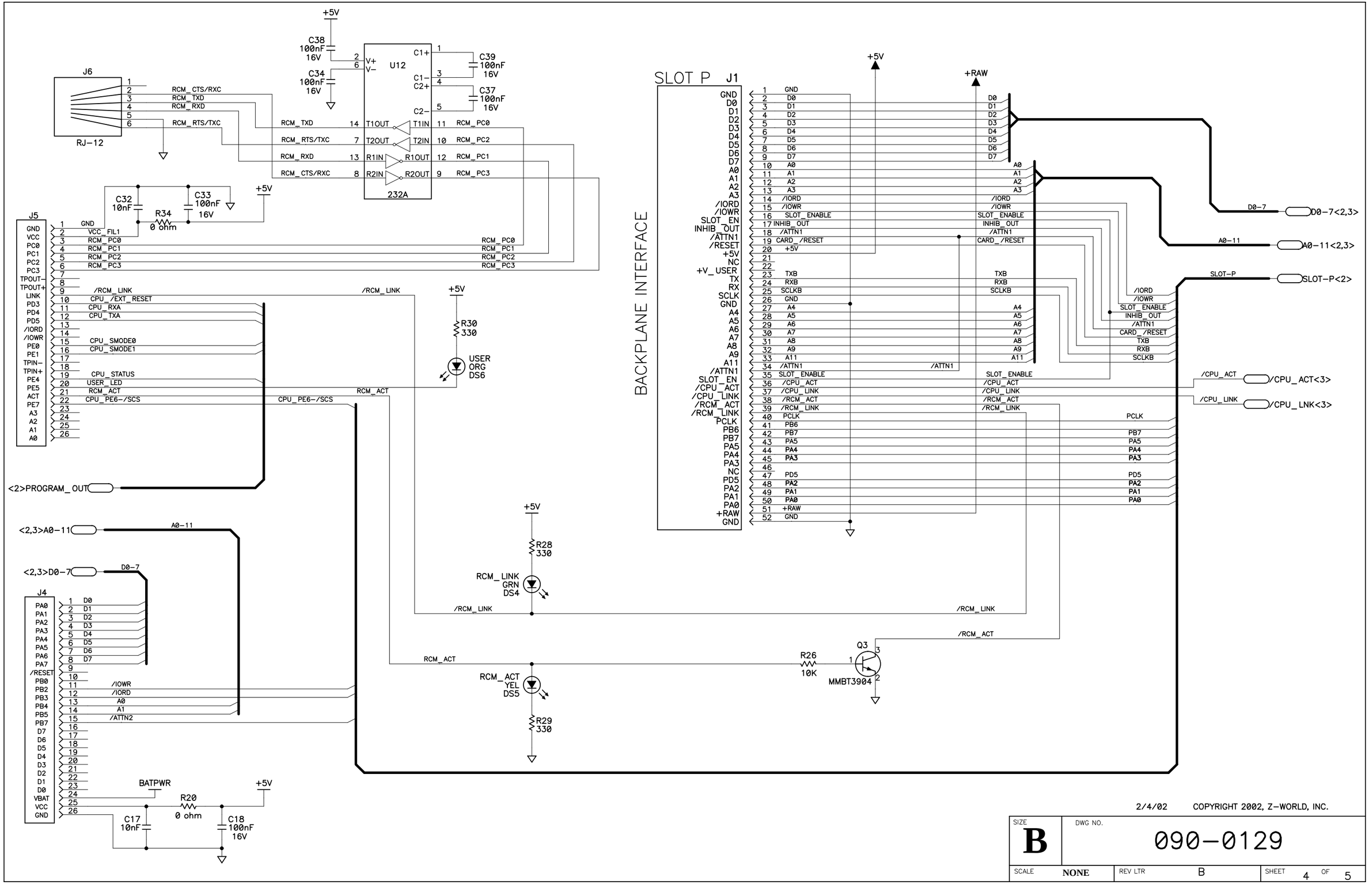
R1 22K
R2 100 1/10W
R3 10K
R6 22K
R8 47K
R11 2K
R12 1M
R13 330k
R14 22M
R24 330
C4 33pF
C6 10nF
C10 22pF
C11 33pF
C12 10nF
C8 15pF
Y1 11.0592MHz
Y2 SMT 32.768KHz
RT1 22k
D1 914
D2 MMBD7000
DS1 RED PWR
Q2 2N7002
U3 74AHC1GU04

J2 RABBIT PROG PORT
R5 100 1/10W
R17 100 1/10W

SIZE B
DWG NO. 090–0129
SCALE NONE
REV LTR B
SHEET 2 OF 5

2/4/02


+5V
C27 100nF 16V
C23 100nF 16V
U10
V+
V-
C1+
C1-
C2+
C2-
C25 100nF 16V
C24 100nF 16V
232A
J3
TXC
RXC
RTS/TXD
CTS/RXD
GND
RS-485+
RS-485-
T1OUT
T2OUT
R1IN
R2IN
T1IN
T2IN
R1OUT
R2OUT
PC2_TXC
TXD
PC3_RXC
CTS
POWER UP STATE
RS-485 DRIVER DISABLED
2 RS-232 PORTS, BOTH 3-WIRE
U6
SP483EN
RS485+
RS485-
PC0_TXD
485_RXD1
RS485_ENABLE
U9:A
74HC04
U8:B
74HC125
U8:A
74HC125
U9:B
74HC04
U8:C
74HC125
U8:D
74HC125
5WIRE/3WIRE
RTS
PC0_TXD
PC1_RXD
RS232/RS485
RS232/RS485<2>
5WIRE/3WIRE
CTS
RS485_ENABLE
JP1
TERMINATOR JUMPER
TO ENABLE, PLACE JUMPERS AT
PINS 1-2 AND 5-6
R22 681 1/10W 1%
R18 220 1/8W
R19 681 1/10W 1%
<2,4>A0-11
A0-11
U14
RLT8019AS*
<2,4>D0-7
D0-7
<2,4>ETHERNET
ETHERNET
R4 10K
PE2
/IORD
/IOWR
PD6
660-0148
R40 200*
R44 10K*
TPIN+
TPIN-
TPOUT+
TPOUT-
J7
RJ-45*
XMIT
RCV
LPF
TX_CT
ERX_CT
ETX_CT
RX_CT
E_TX+
E_TX-
E_RX+
E_RX-
LF1S022
RJ-45 FRONT VIEW
RJ-45 PLUG
TRANSMIT-
TRANSMIT+
RECEIVE-
RECEIVE+
C45 10nF*
C47 10nF*
C50 10nF* 1kV
1kV 10nF* C52
R43 (0 ohm*)
E
ETHERNET
DS2 GRN*
CPU_LINK
R25 330*
/CPU_LINK
/CPU_LINK<4>
/CPU_ACT
/CPU_ACT<4>
R23 10K*
Q1 MMBT3904*
DS3 YELLOW*
CPU_ACT
R27 330*
ACT
R32 (0 ohm*)
R33 1M*
Y3 20.0000MHz*
C31 10pF*
C36 10pF*
X1
X2
JP
LEDBNC
LED0
LED1
LED2
AUI
BCSB
EECS
BD0
BD1
BD2
BD3
BD4
BD5
BD6
BD7
CD+
CD-
RX+
RX-
TX+
TX-
2/4/02
COPYRIGHT 2002, Z-WORLD, INC.
SIZE B
DWG NO. 090-0129
SCALE NONE
REV LTR B
SHEET 3 OF 5



2/4/02 COPYRIGHT 2002, Z-WORLD, INC.

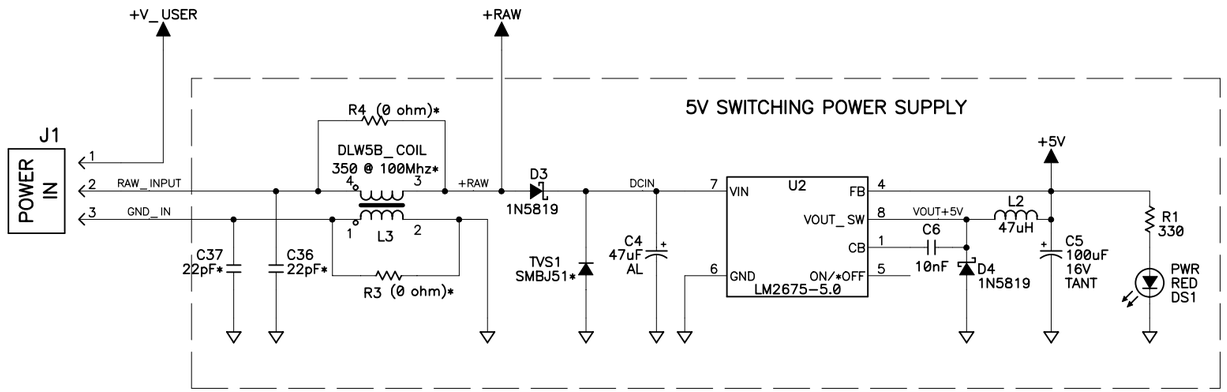
SIZE	DWG NO.	
B	090−0129	
SCALE NONE	REV LTR B	SHEET 4 OF 5

STUFFING TABLE

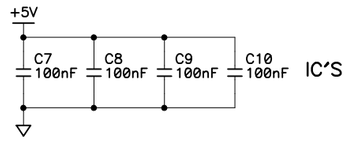
			MODEL	
CIRCUIT		PART	SR9150	SR9160
POWER SUPPLY (PAGE 1)		TVS1	NOT INSTALLED	NOT INSTALLED
		C51	NOT INSTALLED	NOT INSTALLED
		C48	NOT INSTALLED	NOT INSTALLED
		C42	NOT INSTALLED	NOT INSTALLED
		D4, D6	NOT INSTALLED	NOT INSTALLED
		U13	NOT INSTALLED	NOT INSTALLED
		L1	NOT INSTALLED	NOT INSTALLED
SRAM (PAGE 2)		R35, R38	NOT INSTALLED	NOT INSTALLED
	MAIN	U7	128K SRAM	128K SRAM
	SRAM SELECT	JP3	ZERO ohm ACROSS PINS 1–2	ZERO ohm ACROSS PINS 1–2
FLASH (PAGE 2)	FLASH 1	U11	256K FLASH	256K FLASH
	FLASH 1 SELECT	JP4	ZERO ohm ACROSS PINS 1–2	ZERO ohm ACROSS PINS 1–2
	ADDR. BANK SELECT	JP5	ZERO ohm ACROSS PINS 1–2	ZERO ohm ACROSS PINS 1–2
	FLASH 2	U5	256K FLASH	256K FLASH
	FLASH 2 SELECT	JP2	ZERO ohm ACROSS PINS 1–2	ZERO ohm ACROSS PINS 1–2

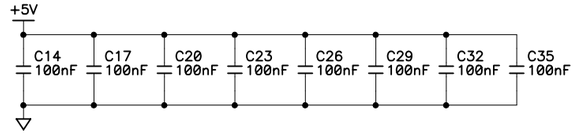
		MODEL	
CIRCUIT	PART	SR9150	SR9160
ETHERNET (PAGE 3)	C50, C52	10nF–1KV	NOT INSTALLED
	C45, C47	10nF	NOT INSTALLED
	C31, C36	10pF	NOT INSTALLED
	C30, C35	100nF–16V	NOT INSTALLED
	C41, C46	100nF–16V	NOT INSTALLED
	C49, C53	100nF–16V	NOT INSTALLED
	Q1	3904 NPN	NOT INSTALLED
	DS2	GREEN LED	NOT INSTALLED
	DS3	YELLOW LED	NOT INSTALLED
	R43	0 ohm	NOT INSTALLED
	R40	200 ohm	NOT INSTALLED
	R33	1M	NOT INSTALLED
	R32	0 ohm	NOT INSTALLED
	R25, R27	330 ohm	NOT INSTALLED
	R23, R44	10K	NOT INSTALLED
	U14	RTL8019AS	NOT INSTALLED
	Y3	20 MHZ CRYSTAL	NOT INSTALLED
	J7	RJ–45	NOT INSTALLED

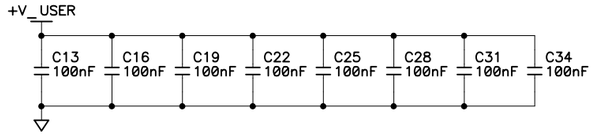
SIZE	DWG NO.	
B	090–0129	
SCALE NONE	REV LTR B	SHEET 5 OF 5



Decoupling Capacitors







NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (V_{CC}), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (V_{CC}).
4. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11833	INITIAL RELEASE	XT	2/15/02	KlS	2/15/02

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION			DEVICE: FILTER CAP REF DES(s)
		GND	+5V	NO CONNECTS	
U2	LM2675-5.0			2,3	
U3	74VHC245	10	20		C7
U4	74VHC245	10	20		C8
U5	74VHC138	8	16		C9
U6	74VHC245	10	20		C10

STUFFING TABLE

	PART	SR9010 BACKPLANE
POWER SUPPLY CIRCUIT (PAGE 1)	R3, R4	ZERO OHM RES
	L3	NOT STUFFED
	C36	NOT STUFFED
	C37	NOT STUFFED
	TVS1	NOT STUFFED
TERMINATION RES (PAGE 3)	R2, R5	NOT STUFFED
LED JUMPER (PAGE 2)	JP1	1-2, 5-6, 9-10,13-14

11FEB02

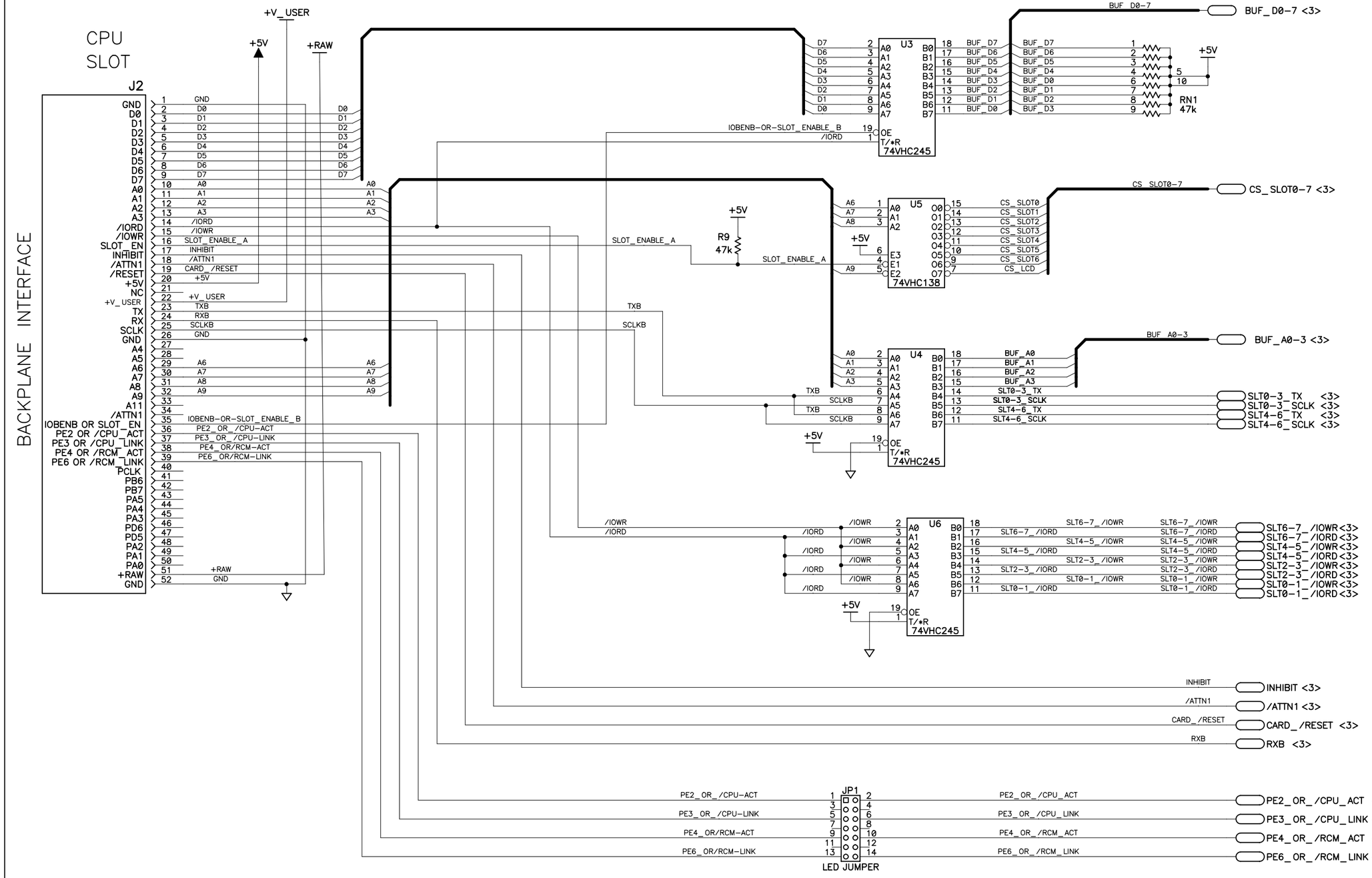
APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:		DRAWING CONTENT:	
		DRAWN BY: (INITIAL RELEASE) XUAN TRUONG	11FEB02
		REVISED BY:	
		APPROVALS: INITIAL RELEASE	
		PROJECT ENGINEER: Xuan Truong	2/15/02
		ENGINEERING MANAGER: R.Matthews	2/15/02
		SIGNATURES	DATE

TITLE: SCHEMATIC DIAGRAM SR9010 7 SLOT BACKPLANE

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

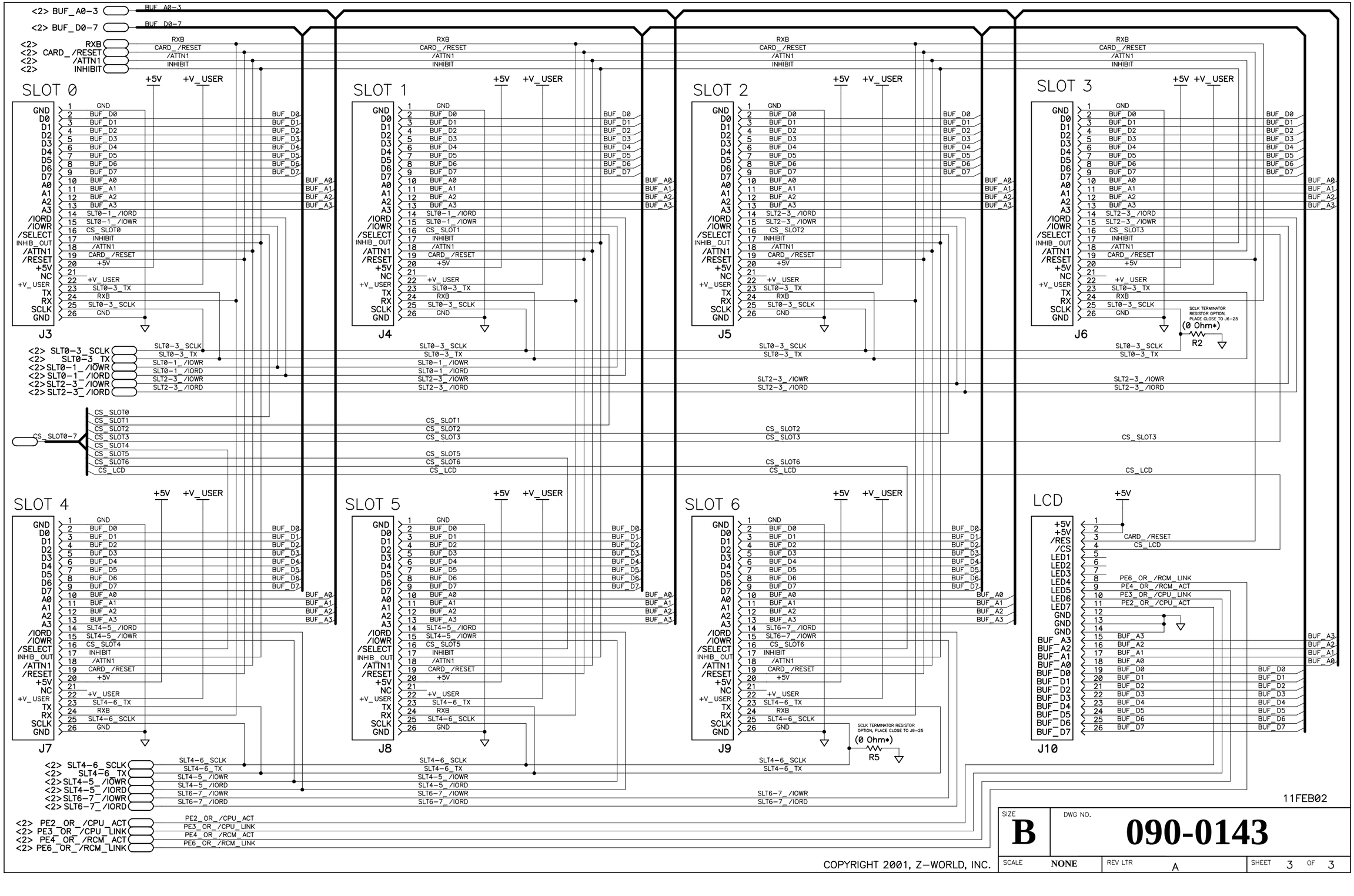
SIZE B | DWG NO. 090-0143

SCALE NONE | RELEASE DATE 2/15/02 | SHEET 1 OF 3

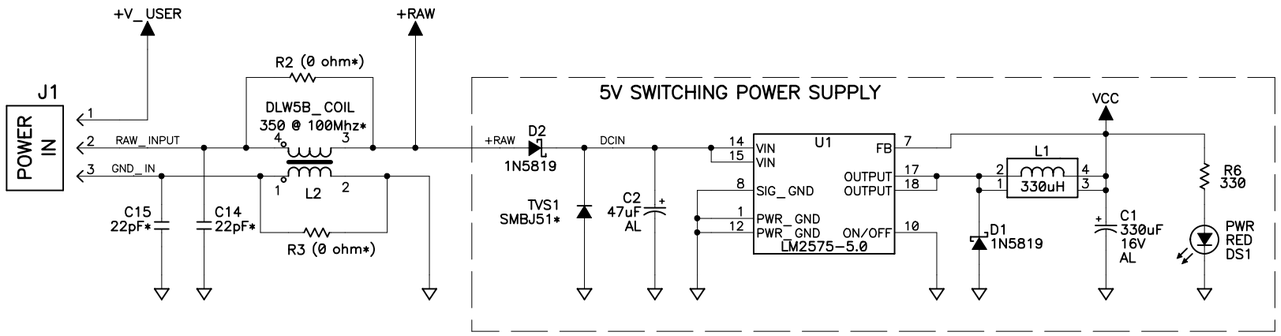


11FEB02

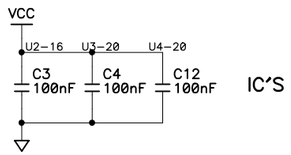
SIZE	DWG NO.	
B	090-0143	
SCALE NONE	REV LTR A	SHEET 2 OF 3

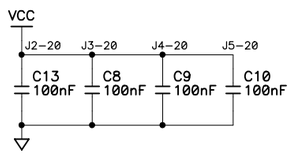

<2> BUF_A0-3
BUF_A0-3
<2> BUF_D0-7
BUF_D0-7
<2> RXB
<2> CARD_/RESET
<2> /ATTN1
<2> INHIBIT
RXB
CARD_/RESET
/ATTN1
INHIBIT
+5V
+V_USER
SLOT 0
SLOT 1
SLOT 2
SLOT 3
GND
D0
D1
D2
D3
D4
D5
D6
D7
A0
A1
A2
A3
/IORD
/IOWR
/SELECT
INHIB_OUT
/ATTN1
/RESET
+5V
NC
+V_USER
TX
RX
SCLK
GND
BUF_D0
BUF_D1
BUF_D2
BUF_D3
BUF_D4
BUF_D5
BUF_D6
BUF_D7
BUF_A0
BUF_A1
BUF_A2
BUF_A3
SLT0-1_/IORD
SLT0-1_/IOWR
CS_SLOT0
CS_SLOT1
CS_SLOT2
CS_SLOT3
INHIBIT
/ATTN1
CARD_/RESET
SLT0-3_TX
RXB
SLT0-3_SCLK
SLT2-3_/IORD
SLT2-3_/IOWR
J3
J4
J5
J6
SCLK TERMINATOR RESISTOR OPTION, PLACE CLOSE TO J6-25
(0 Ohm*)
R2
<2> SLT0-3_SCLK
<2> SLT0-3_TX
<2> SLT0-1_/IOWR
<2> SLT0-1_/IORD
<2> SLT2-3_/IOWR
<2> SLT2-3_/IORD
CS_SLOT0-7
CS_SLOT4
CS_SLOT5
CS_SLOT6
CS_LCD
SLOT 4
SLOT 5
SLOT 6
LCD
SLT4-5_/IORD
SLT4-5_/IOWR
CS_SLOT4
SLT4-6_TX
SLT4-6_SCLK
SLT6-7_/IORD
SLT6-7_/IOWR
CS_SLOT7
RES
/CS
LED1
LED2
LED3
LED4
LED5
LED6
LED7
J7
J8
J9
J10
SCLK TERMINATOR RESISTOR OPTION, PLACE CLOSE TO J9-25
(0 Ohm*)
R5
<2> SLT4-6_SCLK
<2> SLT4-6_TX
<2> SLT4-5_/IOWR
<2> SLT4-5_/IORD
<2> SLT6-7_/IOWR
<2> SLT6-7_/IORD
<2> PE2_OR_/CPU_ACT
<2> PE3_OR_/CPU_LINK
<2> PE4_OR_/RCM_ACT
<2> PE6_OR_/RCM_LINK
PE2_OR_/CPU_ACT
PE3_OR_/CPU_LINK
PE4_OR_/RCM_ACT
PE6_OR_/RCM_LINK
PE6_OR_/RCM_LINK
PE4_OR_/RCM_ACT
PE3_OR_/CPU_LINK
PE2_OR_/CPU_ACT
11FEB02
COPYRIGHT 2001, Z-WORLD, INC.
SIZE
B
DWG NO.
090-0143
SCALE
NONE
REV LTR
A
SHEET
3
OF
3

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11810	INITIAL RELEASE	XT	1/31/02	KIS	1/31/02
B	E11834	SMALL MODIFICATIONS TO ENHANCE READABILITY	XT	2/15/02	KlS	2/15/02



Decoupling Capacitors





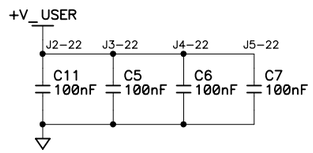


TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION			DEVICE: FILTER CAP REF DES(s)
		GND	+5V	NO CONNECTS	
U1	LM2575-5.0			2-6,9,11,13,16, 19-24	
U2	74VHC138	8	16		C3
U3	74VHC245	10	20		C4
U4	74VHC245	10	20		C12

NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).
4. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.

STUFFING TABLE

	PART	SR9050 BACKPLANE
POWER SUPPLY CIRCUIT (PAGE 1)	R2, R3	ZERO OHM RES
	L2	NOT STUFFED
	C14	NOT STUFFED
	C15	NOT STUFFED
	TVS1	NOT STUFFED
TERMINATION RES (PAGE 3)	R4	NOT STUFFED
LED JUMPER (PAGE 2)	JP1	1-2, 5-6, 9-10,13-14

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

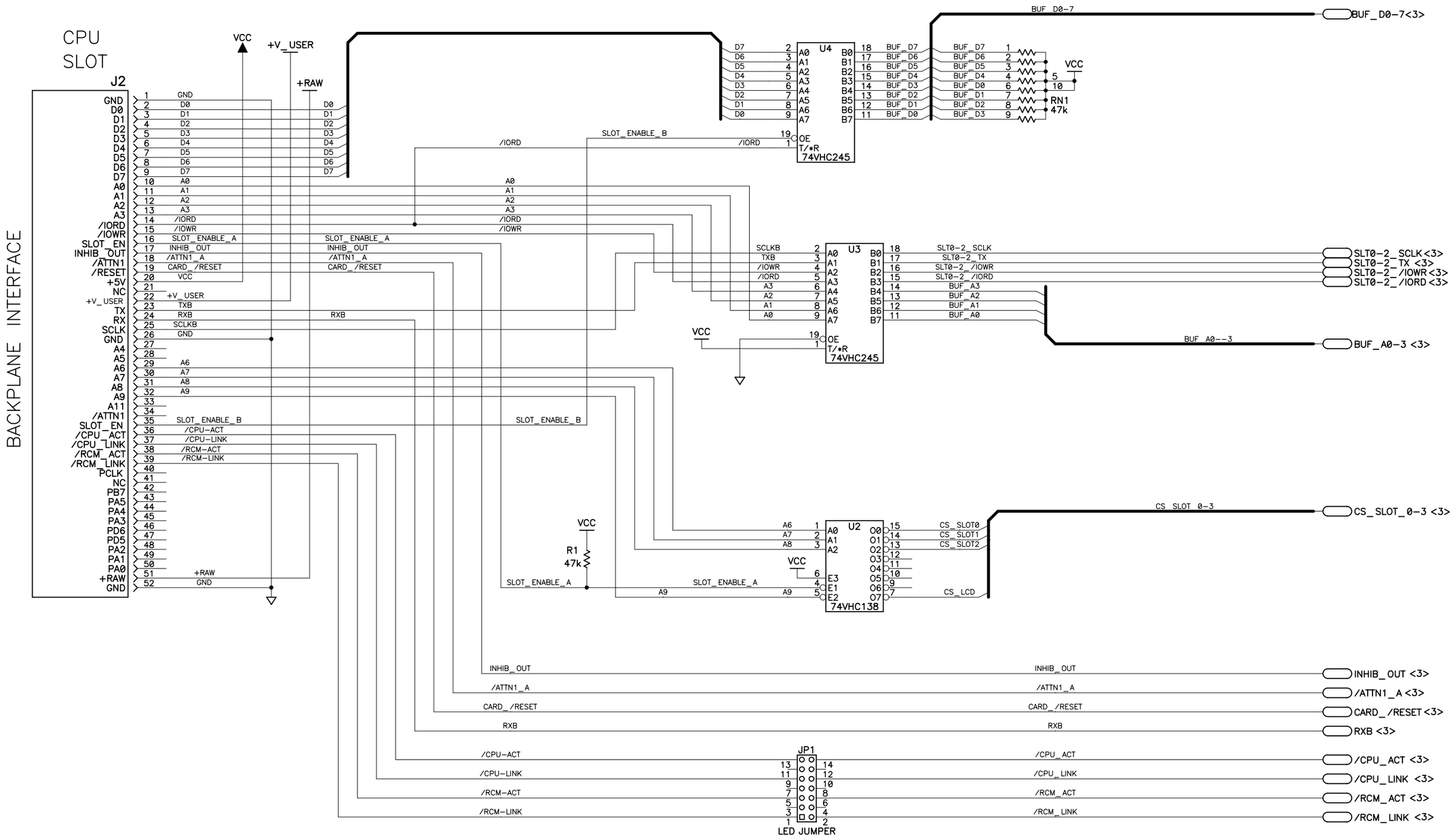
DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) XUAN TRUONG	11/30/01
REVISED BY: K.SCHALLER	2/12/02
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Xuan Truong	1/31/02
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	DATE

TITLE: SCHEMATIC DIAGRAM SR9050 3 SLOT BACKPLANE

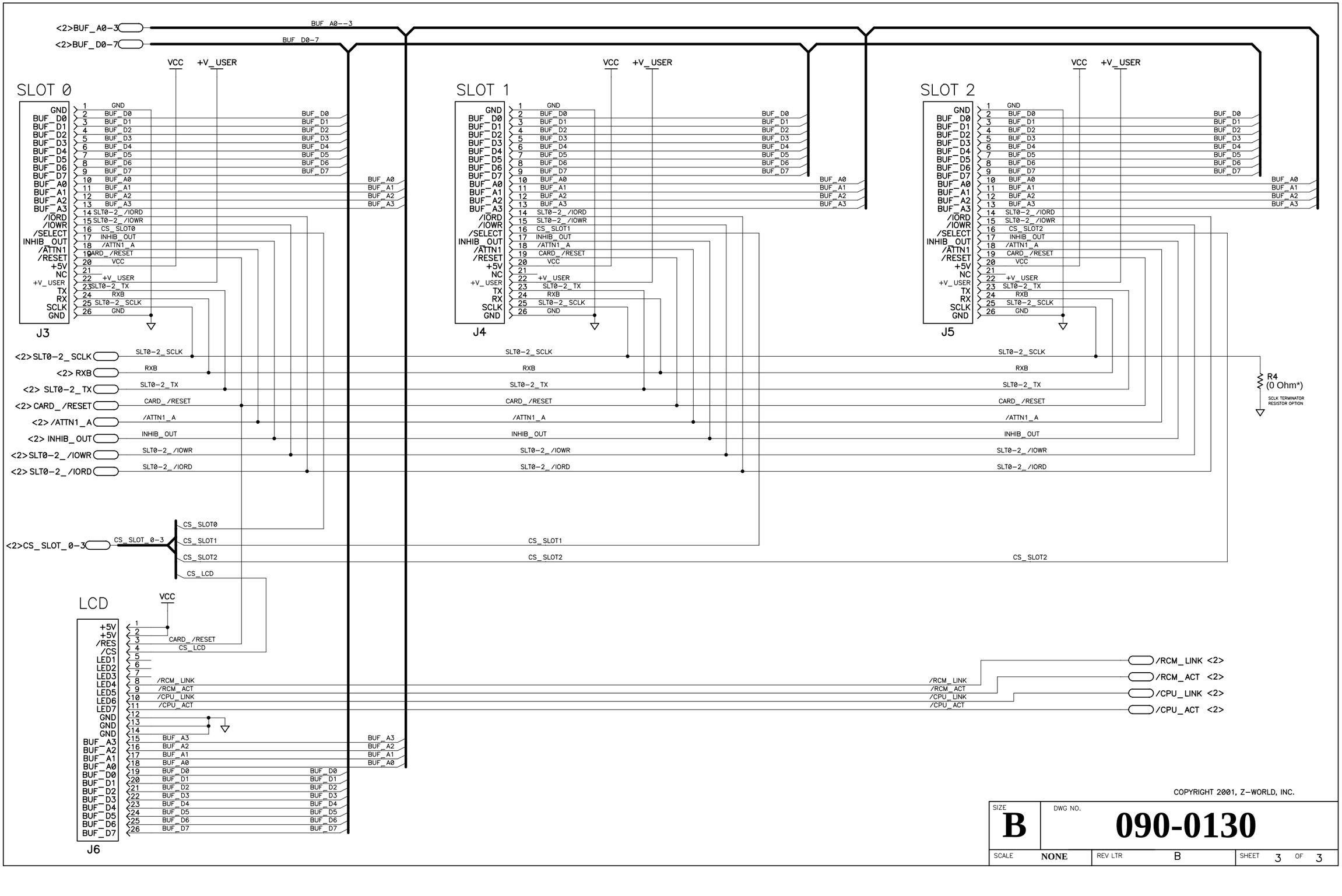
Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE B | DWG NO. 090-0130

SCALE NONE | RELEASE DATE 1/31/02 | SHEET 1 OF 3



SIZE	DWG NO.	
B	090-0130	
SCALE NONE	REV LTR B	SHEET 2 OF 3



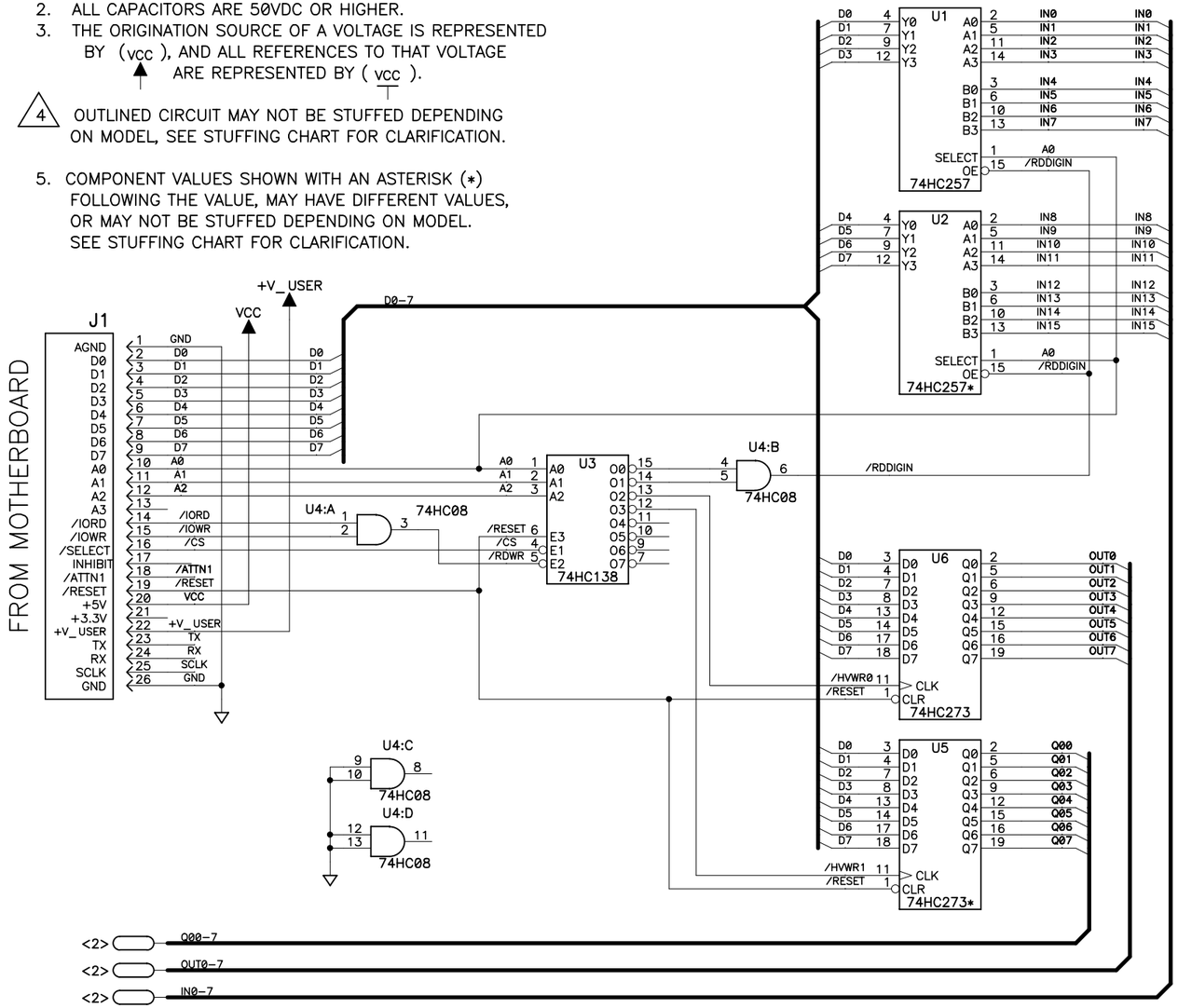
SIZE	DWG NO.	
B	090-0130	
SCALE NONE	REV LTR B	SHEET 3 OF 3

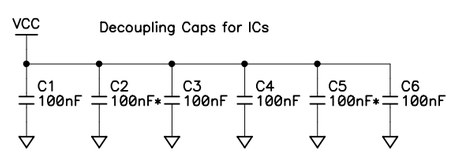
NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

4 OUTLINED CIRCUIT MAY NOT BE STUFFED DEPENDING ON MODEL, SEE STUFFING CHART FOR CLARIFICATION.

5. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES, OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.





REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11217	INITIAL RELEASE, TRACKS A/W @ REV-A	DM	11/20/00	KIS	11/17/01
B	E11564	CHANGED PE7 TO OUTPUT INHIBIT	DM	28JUN01	KIS	7/17/01

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION					DEVICE: FILTER CAP
		AGND	GND	VCC	-5V	NO CONNECTS	REF DES(s)
U1	74HC257		8	16			C1
U2	74HC257		8	16			C2
U3	74HC138		8	16			C3
U4	74HC08		7	14			C4
U5	74HC273		10	20			C5
U6	74HC273		10	20			C6

STUFFING TABLE

	CIRCUIT		PART	16 INPUT 8 OUTPUT	8 INPUT 16 OUTPUT	8 INPUT 8 OUTPUT
				MODEL		
				SR9200	SR9210	SR9220
	FPWR SELECT		JP1	ZERO ohm ACROSS PINS 2-3	ZERO ohm ACROSS PINS 2-3	ZERO ohm ACROSS PINS 2-3
BANK 0	INPUT LEVEL SELECT		JP2	ZERO ohm ACROSS PINS 1-2	ZERO ohm ACROSS PINS 1-2	ZERO ohm ACROSS PINS 1-2
BANK 2	HIGH VOLTAGE SINKING DRIVERS	CURRENT DRIVER	R36-43 C28-35 D28-35 Q28-35	NOT INSTALLED	INSTALLED	NOT INSTALLED
		DRIVER CHIP	U5 C5	NOT INSTALLED	INSTALLED	NOT INSTALLED
	DIGITAL INPUTS	INPUT CONDITIONING	R60-67 C44-51	INSTALLED	NOT INSTALLED	NOT INSTALLED
		DRIVER CHIP	U2 C2	INSTALLED	NOT INSTALLED	NOT INSTALLED
	DIGITAL I/O LEVEL		JP3 1-2 R68-75	INSTALLED	NOT INSTALLED	NOT INSTALLED
BANK 1	DIGITAL OUTPUT LEVEL		R28-35	NOT INSTALLED	NOT INSTALLED	NOT INSTALLED

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) RM	29JUL99
REVISED BY: DM	26JUN01
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	DATE

TITLE

SCHEMATIC DIAGRAM SR9200 SERIES DIGITAL I/O BOARD

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE B | DWG NO. 090-0101

SCALE NONE | RELEASE DATE | SHEET 1 OF 2

OUT0–7
Q00–7
IN0–7

USER INTERFACE

J2

GND	1	GND	
DIGIN0	2	DIGIN0	DIGIN0
DIGIN1	3	DIGIN1	DIGIN1
DIGIN2	4	DIGIN2	DIGIN2
DIGIN3	5	DIGIN3	DIGIN3
GND	6	GND	
K	7	K	
DIGIN4	8	DIGIN4	DIGIN4
DIGIN5	9	DIGIN5	DIGIN5
DIGIN6	10	DIGIN6	DIGIN6
DIGIN7	11	DIGIN7	DIGIN7
GND	12	GND	
I/O8	13	I/O8	I/O8
I/O9	14	I/O9	I/O9
I/O10	15	I/O10	I/O10
I/O11	16	I/O11	I/O11
GND	17	GND	
I/O12	18	I/O12	I/O12
I/O13	19	I/O13	I/O13
I/O14	20	I/O14	I/O14
I/O15	21	I/O15	I/O15
HVOUT0	22	HVOUT0	HVOUT0
HVOUT1	23	HVOUT1	HVOUT1
HVOUT2	24	HVOUT2	HVOUT2
GND	25	GND	
HVOUT3	26	HVOUT3	HVOUT3
HVOUT4	27	HVOUT4	HVOUT4
K	28	K	
HVOUT5	29	HVOUT5	HVOUT5
HVOUT6	30	HVOUT6	HVOUT6
HVOUT7	31	HVOUT7	HVOUT7
	32		
GND	33	GND	
	34		

Install a 0 Ohm Resistor on
JP1:1–2 for K (Customer provides Power Source)
JP1:2–3 for +V_USER (Power from Z–Bus)

+V_USER
JP1*
FPWR
K

BANK 0
DIGITAL INPUTS

BANK 2
FACTORY CONFIGURABLE
DIGITAL INPUTS
4

Install 0 Ohm Resistor on
JP2:1–2 for Pull–Ups
JP2:2–3 for Pull–Downs

VCC
JP2*

IN0	R52 22k	C36 10nF	DIGIN0
IN1	R53 22k	C37 10nF	DIGIN1
IN2	R54 22k	C38 10nF	DIGIN2
IN3	R55 22k	C39 10nF	DIGIN3
IN4	R56 22k	C40 10nF	DIGIN4
IN5	R57 22k	C41 10nF	DIGIN5
IN6	R58 22k	C42 10nF	DIGIN6
IN7	R59 22k	C43 10nF	DIGIN7

DIGIN0 R44
DIGIN1 R45
DIGIN2 R46
DIGIN3 R47
DIGIN4 R48
DIGIN5 R49
DIGIN6 R50
DIGIN7 R51
10k

IN8	R67 22k	C51 10nF
IN9	R66 22k	C50 10nF
IN10	R65 22k	C49 10nF
IN11	R64 22k	C48 10nF
IN12	R63 22k	C47 10nF
IN13	R62 22k	C46 10nF
IN14	R61 22k	C45 10nF
IN15	R60 22k	C44 10nF

I/O8 R67
I/O9 R66
I/O10 R65
I/O11 R64
I/O12 R63
I/O13 R62
I/O14 R61
I/O15 R60

DIGIIN0–7
I/O8–15
HVOUT0–7

BANK 2
FACTORY CONFIGURABLE
HIGH VOLTAGE SINKING DRIVERS
4

Q00	R43 536 1%	Q35 MMBT4401	D35 914	C35 100nF
Q01	R42 536 1%	Q34 MMBT4401	D34 914	C34 100nF
Q02	R41 536 1%	Q33 MMBT4401	D33 914	C33 100nF
Q03	R40 536 1%	Q32 MMBT4401	D32 914	C32 100nF
Q04	R39 536 1%	Q31 MMBT4401	D31 914	C31 100nF
Q05	R38 536 1%	Q30 MMBT4401	D30 914	C30 100nF
Q06	R37 536 1%	Q29 MMBT4401	D29 914	C29 100nF
Q07	R36 536 1%	Q28 MMBT4401	D28 914	C28 100nF

FPWR
I/O8
I/O9
I/O10
I/O11
I/O12
I/O13
I/O14
I/O15

DIGITAL I/O
LEVEL
4

Install 0 Ohm Resistor on
JP3:1–2 for Pull–Ups
JP3:2–3 for Pull–Downs

VCC
JP3

I/O8	R75
I/O9	R74
I/O10	R73
I/O11	R72
I/O12	R71
I/O13	R70
I/O14	R69
I/O15	R68

10k

BANK 1
HIGH VOLTAGE SINKING DRIVERS

OUT0	R27 536 1%	Q27 MMBT4401	D27 914	C27 100nF
OUT1	R26 536 1%	Q26 MMBT4401	D26 914	C26 100nF
OUT2	R25 536 1%	Q25 MMBT4401	D25 914	C25 100nF
OUT3	R24 536 1%	Q24 MMBT4401	D24 914	C24 100nF
OUT4	R23 536 1%	Q23 MMBT4401	D23 914	C23 100nF
OUT5	R22 536 1%	Q22 MMBT4401	D22 914	C22 100nF
OUT6	R21 536 1%	Q21 MMBT4401	D21 914	C21 100nF
OUT7	R20 536 1%	Q20 MMBT4401	D20 914	C20 100nF

FPWR
HVOUT0
HVOUT1
HVOUT2
HVOUT3
HVOUT4
HVOUT5
HVOUT6
HVOUT7

DIGITAL OUTPUT
LEVELS
4

VCC

HVOUT0	R35 10k*
HVOUT1	R34 10k*
HVOUT2	R33 10k*
HVOUT3	R32 10k*
HVOUT4	R31 10k*
HVOUT5	R30 10k*
HVOUT6	R29 10k*
HVOUT7	R28 10k*

26JUN01

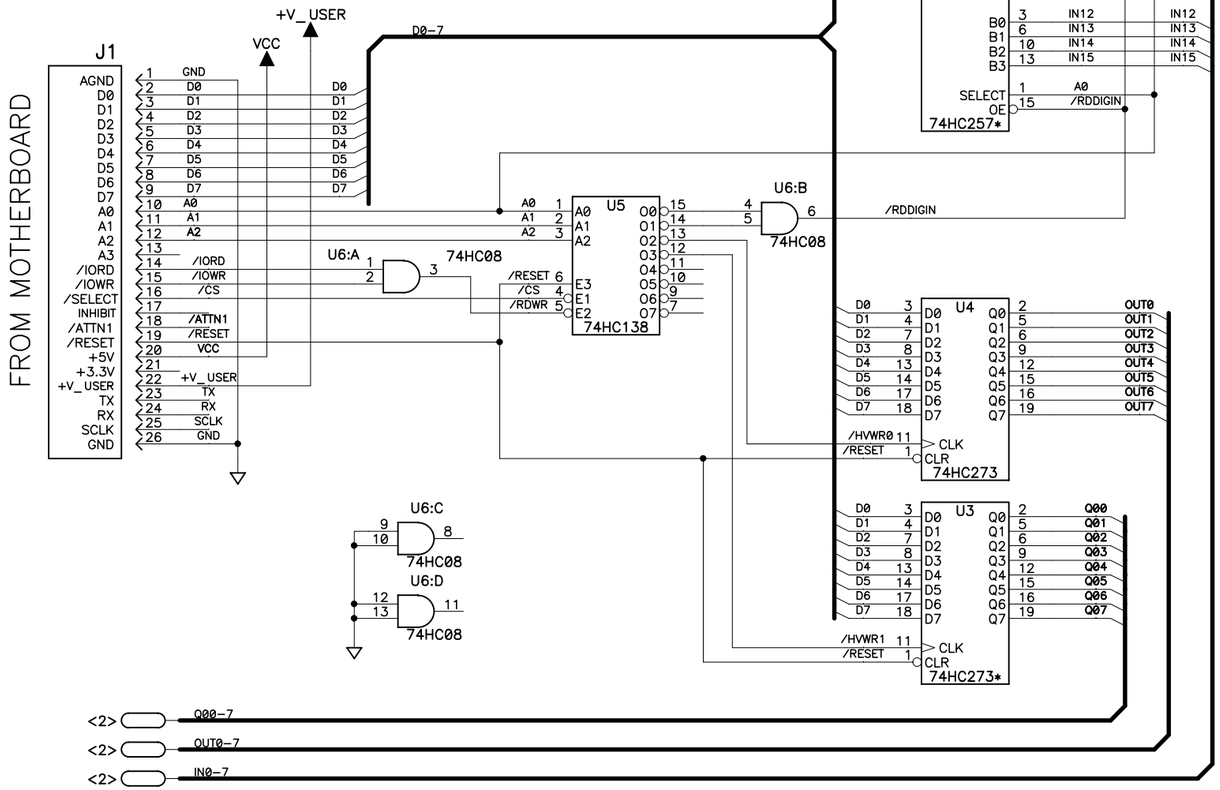
SIZE	DWG NO.	
B	090–0101	
SCALE NONE	REV LTR B	SHEET 2 OF 2

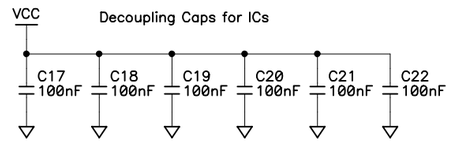
NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/16W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

4 OUTLINED CIRCUIT MAY NOT BE STUFFED DEPENDING ON MODEL, SEE STUFFING CHART FOR CLARIFICATION.

5. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES, OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.





REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11383	INITIAL RELEASE, TRACKS A/W @ REV-A	XT	2/1/01	KIS	2/1/01
B	E11564	CHANGED PE7 TO OUTPUT INHIBIT	DM	28JUN01	KIS	7/17/01

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION					DEVICE: FILTER CAP REF DES(s)
		AGND	GND	VCC	-5V	NO CONNECTS	
U1	74HC257		8	16			C17
U2	74HC257		8	16			C18
U3	74HC273		10	20			C19
U4	74HC273		10	20			C20
U5	74HC138		8	16			C21
U6	74HC08		7	14			C22

STUFFING TABLE

CIRCUIT			PART	MODEL: SR9205 16 INPUT 8 OUPUT	MODEL: SR9215 8 INPUT 16 OUTPUT	MODEL: SR9225 8 INPUT 8 OUTPUT
FPWR SELECT			JP3	ZERO OHM ACROSS PINS 2-3	ZERO OHM ACROSS PINS 2-3	ZERO OHM ACROSS PINS 2-3
BANK 2	DIGITAL I/O LEVEL		JP2 R25-32	ZERO ohm ACROSS PINS 1-2 10k OHMS	ZERO ohm ACROSS PINS 2-3 100k OHMS	NOT INSTALLED
	HIGH VOLTAGE SOURCING DRIVERS	CURRENT DRIVER	R72-95 C48-55 D9-16 Q17-32	NOT INSTALLED	INSTALLED	NOT INSTALLED
		DRIVER CHIP	U3 C19	NOT INSTALLED	INSTALLED	NOT INSTALLED
	DIGITAL INPUTS	INPUT CONDITIONING	R17-24 C9-16	INSTALLED	NOT INSTALLED	NOT INSTALLED
		DRIVER CHIP	U2 C18	INSTALLED	NOT INSTALLED	NOT INSTALLED
INPUT LEVEL SELECT			JP1	ZERO OHM ACROSS PINS 1-2	ZERO OHM ACROSS PINS 1-2	ZERO OHM ACROSS PINS 1-2

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:		
DRAWN BY: (INITIAL RELEASE)	KAH	09NOV00
REVISED BY:	DM	28JUN01
APPROVALS: INITIAL RELEASE		
PROJECT ENGINEER:	Darren Musgrove	7/20/01
ENGINEERING MANAGER:	R.Matthews	2/15/02
SIGNATURES		DATE

TITLE

SCHEMATIC DIAGRAM SR92x5 SERIES SOURCE OUTPUT BOARD

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE B | DWG NO. 090-0118 | SCALE NONE | RELEASE DATE | SHEET 1 OF 2

USER INTERFACE

OUT0–7
Q00–7
IN0–7

Install a 0 Ohm Resistor on
JP3:1–2 for K (Customer provides Power Source)
JP3:2–3 for +V_USER (Power from Z–Bus)

+V_USER
JP3*
FPWR

J2

BANK 0
DIGITAL INPUTS

BANK 2
FACTORY CONFIGURABLE
DIGITAL INPUTS

BANK 2
FACTORY CONFIGURABLE
HIGH VOLTAGE SOURCING DRIVERS

BANK 1
HIGH VOLTAGE SOURCING DRIVERS

Install 0 Ohm Resistor on
JP1, JP2 :1–2 for Pull–Ups
JP1, JP2 :2–3 for Pull–Downs

10k for Pull–Ups
100k for Pull–Downs

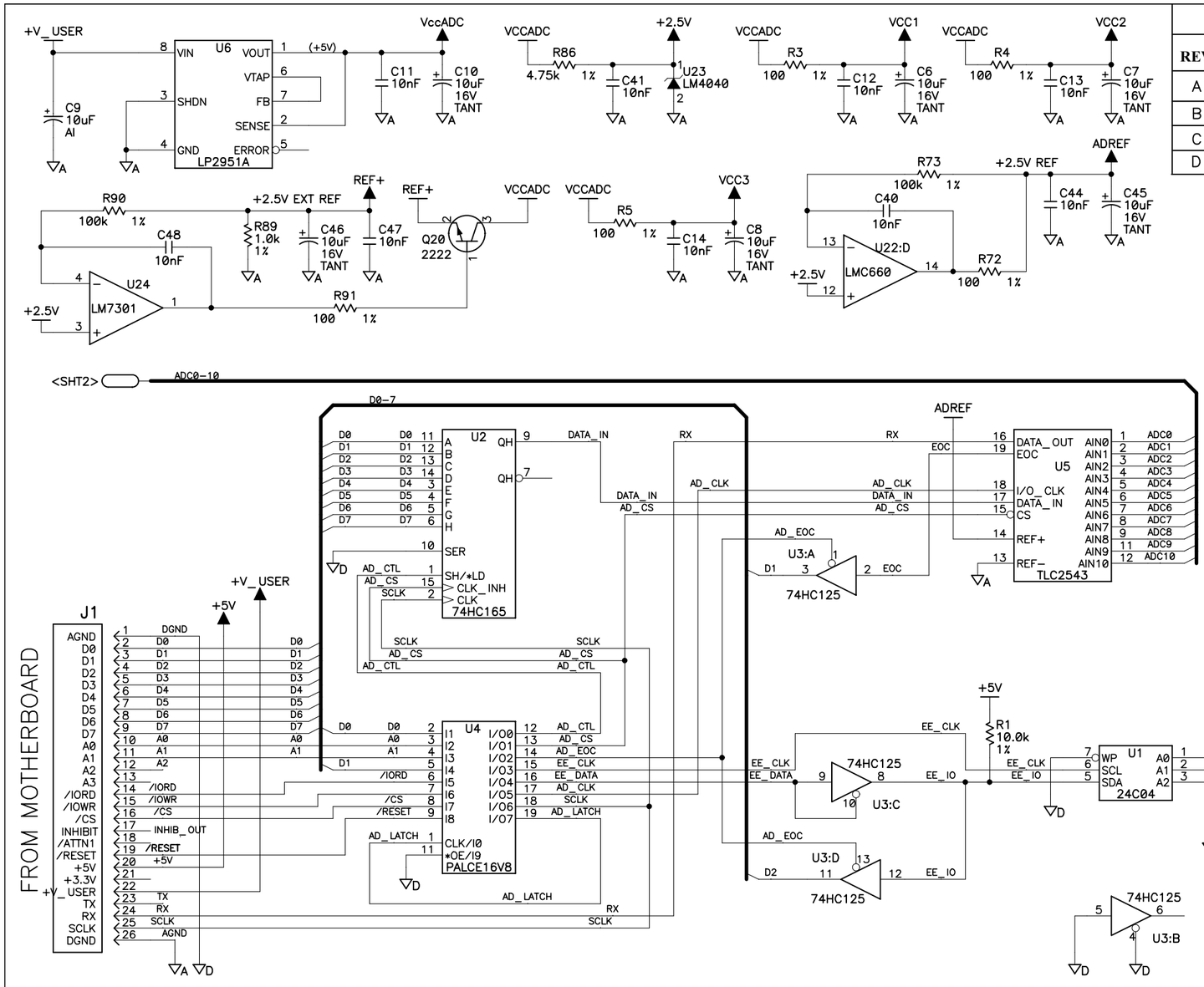
*JP1
*JP2
DIGITAL I/O LEVEL

DIGITAL OUTPUT LEVELS

DIGIIN0–7
I/O8–15
HVOUT0–7

28JUN01

SIZE	DWG NO.	
B	090–0118	
SCALE NONE	REV LTR B	SHEET 2 OF 2



REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	----	PROTOTYPE RELEASE, TRACKS A/W @ REV-A	----	----	----	----
B	E11217	INITIAL RELEASE TRACK A/W @ REV-B	DM	20NOV00	KIS	11/20/01
C	E11453	CONNECTED /IOWR TO U4, UPDATED FOR SR9320	DM	20MAR01	KIS	3/20/01
D	E11564	CHANGED PE7 TO OUTPUT INHIBIT	DM	28JUN01	KIS	7/17/01

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION									DEVICE: FILTER CAP REF DES(s)
		AGND	DGND	+5V	VCC1	VCC2	VCC3	ADREF	VCCADC	NC	
U1	24C04		4	8							C1
U2	74HC165		10	20						1,6,11,16	C2
U3	74HC125		7	14							C3
U4	PALCE16V8		10	20							C4
U5	TLC2543	10							20		C5, C44
U6	LP2951A										
U20	LMC660	11			4						C12
U21	LMC660	11				4					C13
U22	LMC660	11					4				C14
U23	LM4040										C41
U24	LM7301	2							5		C11

STUFFING TABLE

	CIRCUIT	PART	MODEL SR9300	MODEL SR9310	MODEL SR9320
A/D INPUT CONDITIONING	GAIN RESISTOR	R32,39,40,47,48,55, R56,63,64,71,77	23.7k 1%	12.1k 1%	60.4k 1%
	BIAS RESISTOR	R34,37,42,45,50,53 R58,61,66,69,75	39.2k 1%	8.06k 1%	32.4k 1%
	EXCITATION SELECT OR 4-20mA SELECT	JP2,JP4,JP6,JP8 JP10,JP12,JP13 JP15,JP17,JP19 JP21	NOT INSTALLED	NOT INSTALLED	INSTALLED 2-3
	EXCITATION VOLTAGE SELECT	JP1,JP3,JP5,JP7 JP9,JP11,JP14, JP16,JP18,JP20 JP22	NOT INSTALLED	NOT INSTALLED	NOT INSTALLED

NOTES: UNLESS OTHERWISE SPECIFIED;

1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

4 OUTLINED CIRCUIT MAY NOT BE STUFFED DEPENDING ON MODEL, SEE STUFFING CHART FOR CLARIFICATION.

5. COMPONENT VALUES SHOWN WITH AN ASTERISK (*) FOLLOWING THE VALUE, MAY HAVE DIFFERENT VALUES, OR MAY NOT BE STUFFED DEPENDING ON MODEL. SEE STUFFING CHART FOR CLARIFICATION.

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:

DRAWN BY: (INITIAL RELEASE) RAF	18JUN99
REVISED BY: DM	20JUN01
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	7/20/01
SIGNATURES	DATE

TITLE SCHEMATIC DIAGRAM SR9300 SERIES 12-BIT 11-CHANNEL A/D CONVERTER

SIZE B

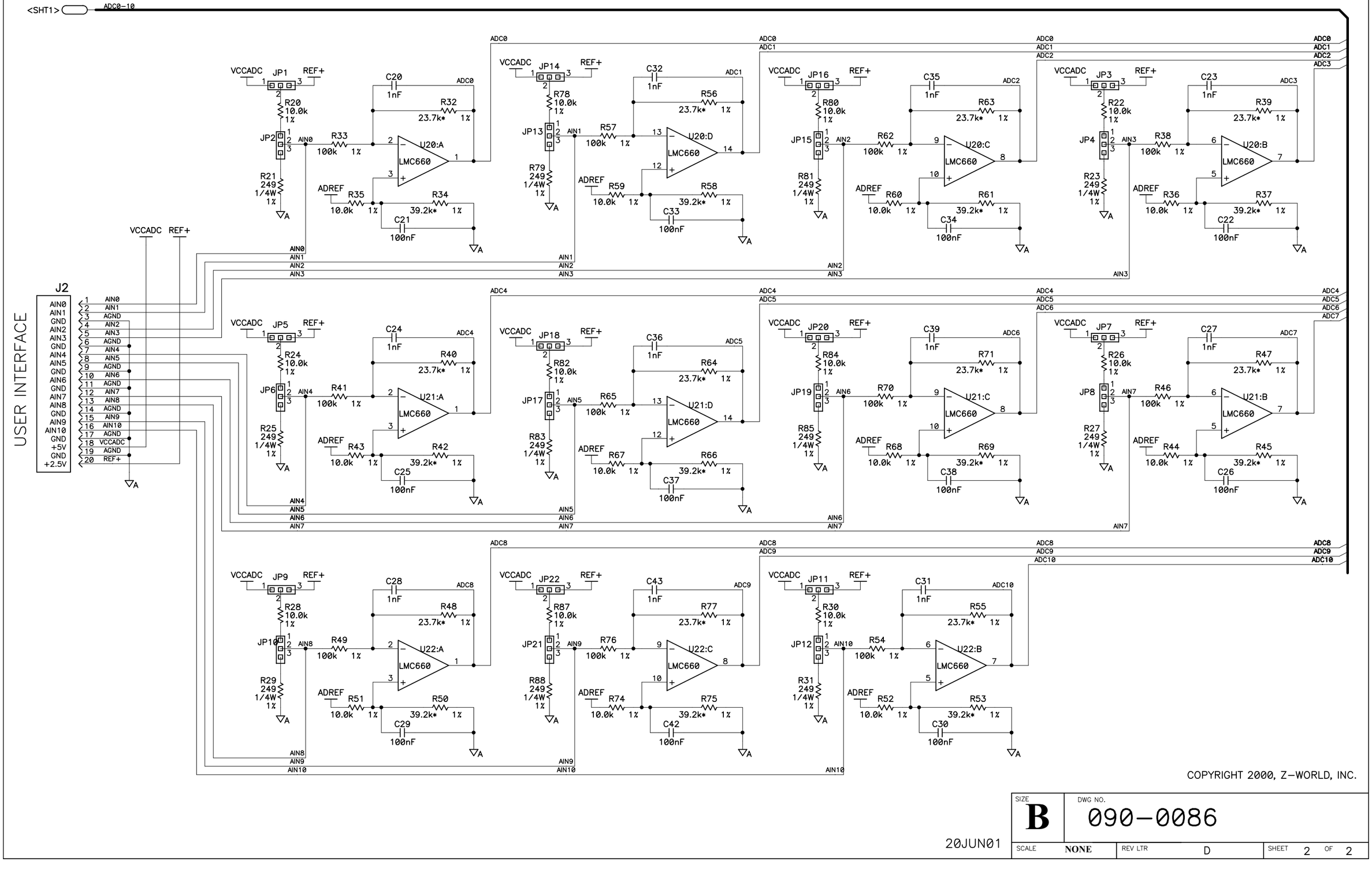
DWG NO. 090-0086

SCALE NONE

RELEASE DATE

SHEET 1 OF 2

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616



20JUN01

SIZE	DWG NO.	
B	090-0086	
SCALE NONE	REV LTR D	SHEET 2 OF 2

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11276	INITIAL RELEASE TO PRODUCTION	J. FENG	4/18/01	KIS	4/18/01
B	E11564	CHANGED PE7 TO OUTPUT INHIBIT, J1, J2 FORMAT	DM	28JUN01	KIS	7/17/01

NOTES: UNLESS OTHERWISE SPECIFIED;
1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) J. FENG	3/1/2001
REVISED BY: D.MUSGROVE	06/28/01
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: J. FENG	4/18/2001
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	DATE

TITLE: SCHEMATIC DIAGRAM SR9400 Digital to Analog Converter

ZWORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE: B | DWG NO. 090-0121 | SCALE NONE | RELEASE DATE | SHEET 1 OF 2

STUFFING TABLE

PART	SR9400 0 – +10.24V D/A CONVERTER	SR9410 −10.24 – +10.24V D/A CONVERTER	SR9420 4 – 20mA D/A CONVERTER
R63	7.87K 1% 0603	7.87K 1% 0603	15.8K 1% 0603
R64	2.62K 1% 0603	7.87K 1% 0603	10.7K 1% 0603
R65	2.80K 1% 0603	2.80K 1% 0603	0 ohm 0603
R66	2.00K 1% 0603	2.00K 1% 0603	NOT INSTALLED
R7	137K 1% 0603	137K 1% 0603	165K 1% 0603
R8	137K 1% 0603	137K 1% 0603	165K 1% 0603
R11	137K 1% 0603	137K 1% 0603	165K 1% 0603
R12	137K 1% 0603	137K 1% 0603	165K 1% 0603
R55	137K 1% 0603	137K 1% 0603	165K 1% 0603
R56	137K 1% 0603	137K 1% 0603	165K 1% 0603
R59	137K 1% 0603	137K 1% 0603	165K 1% 0603
R60	137K 1% 0603	137K 1% 0603	165K 1% 0603
R17	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R18	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R19	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R20	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R27	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R28	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R29	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R30	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R43	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R44	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R45	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R46	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R49	825K 1% 0603	825K 1% 0603	8.25K 1% 0603
R50	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R51	NOT INSTALLED	NOT INSTALLED	8.25K 1% 0603
R52	825K 1% 0603	825K 1% 0603	8.25K 1% 0603

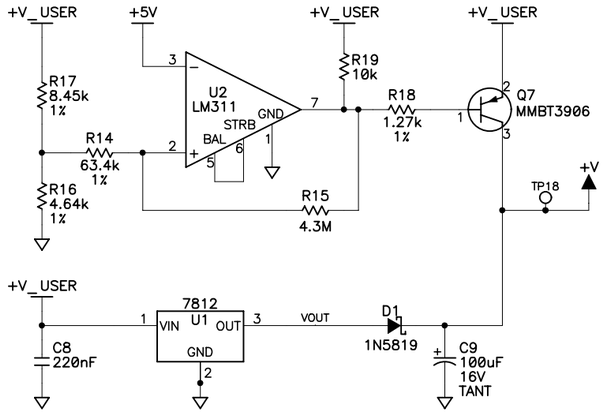
PART	SR9400 0 – +10.24V D/A CONVERTER	SR9410 −10.24 – +10.24V D/A CONVERTER	SR9420 4 – 20mA D/A CONVERTER
R15	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R22	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R25	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R31	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R42	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R47	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R48	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R53	NOT INSTALLED	NOT INSTALLED	0 ohm 0603
R16	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R21	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R26	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R32	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R35	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R36	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R39	0 ohm 0603	0 ohm 0603	NOT INSTALLED
R40	0 ohm 0603	0 ohm 0603	NOT INSTALLED
C13	0 ohm 0805	100nF 0805	0 ohm 0805
C21	0 ohm 0805	100nF 0805	0 ohm 0805
C22	0 ohm 0805	100nF 0805	0 ohm 0805
U9	NOT INSTALLED	LM2675−12, IC	NOT INSTALLED
D1	NOT INSTALLED	1N5819, DIODE	NOT INSTALLED
D3	NOT INSTALLED	1N5819, DIODE	NOT INSTALLED
D4	NOT INSTALLED	1N5819, DIODE	NOT INSTALLED
C11	NOT INSTALLED	330 uF 16V CAP	NOT INSTALLED
C12	NOT INSTALLED	10nF 0805	NOT INSTALLED
C14	NOT INSTALLED	47uF 50V CAP	NOT INSTALLED
L2	NOT INSTALLED	100uH, IND	NOT INSTALLED
R14	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R23	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R24	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R33	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R34	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R37	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R38	100 ohm 0603	100 ohm 0603	10.0 ohm 0603
R41	100 ohm 0603	100 ohm 0603	10.0 ohm 0603

POWER TABLE

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION						DEVICE: FILTER CAP REF DES(s)
		AGND	GND	+5V	+12V	−12V	NO CONNECTS	
U8	24C04		4	8				C10
U10	LM324A				4	11		C19, C21
U6	74VHC574		10	20				C8
U2	TLV5618	5		8				C2
U7	74VHC125		7	14				C9
U3	TLV5618	5		8				C3
U4	TLV5618	5		8				C4
U11	LM324A				4	11		C20, C22
U5	TLV5618	5		8				C5

SIZE	DWG NO.	
B	090−0121	
SCALE NONE	REV LTR B	SHEET 2 OF 2

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11217	INITIAL RELEASE	DM	11/20/01	KIS	7/17/01
B	E11564	CHANGED PE7 TO OUTPUT INHIBIT	DM	28JUN01	KIS	7/17/01
C	E11715	CORRECTED REL2COM TO REL2NO ON J1 PIN 14.	XT	2/15/02	KlS	2/15/02



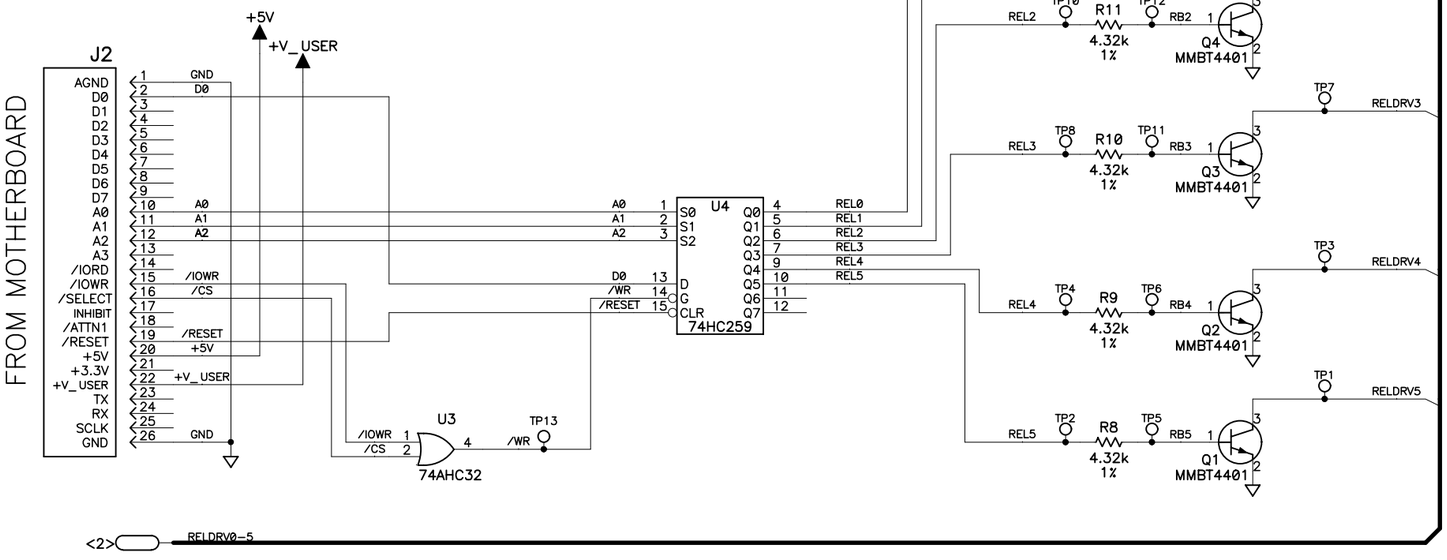


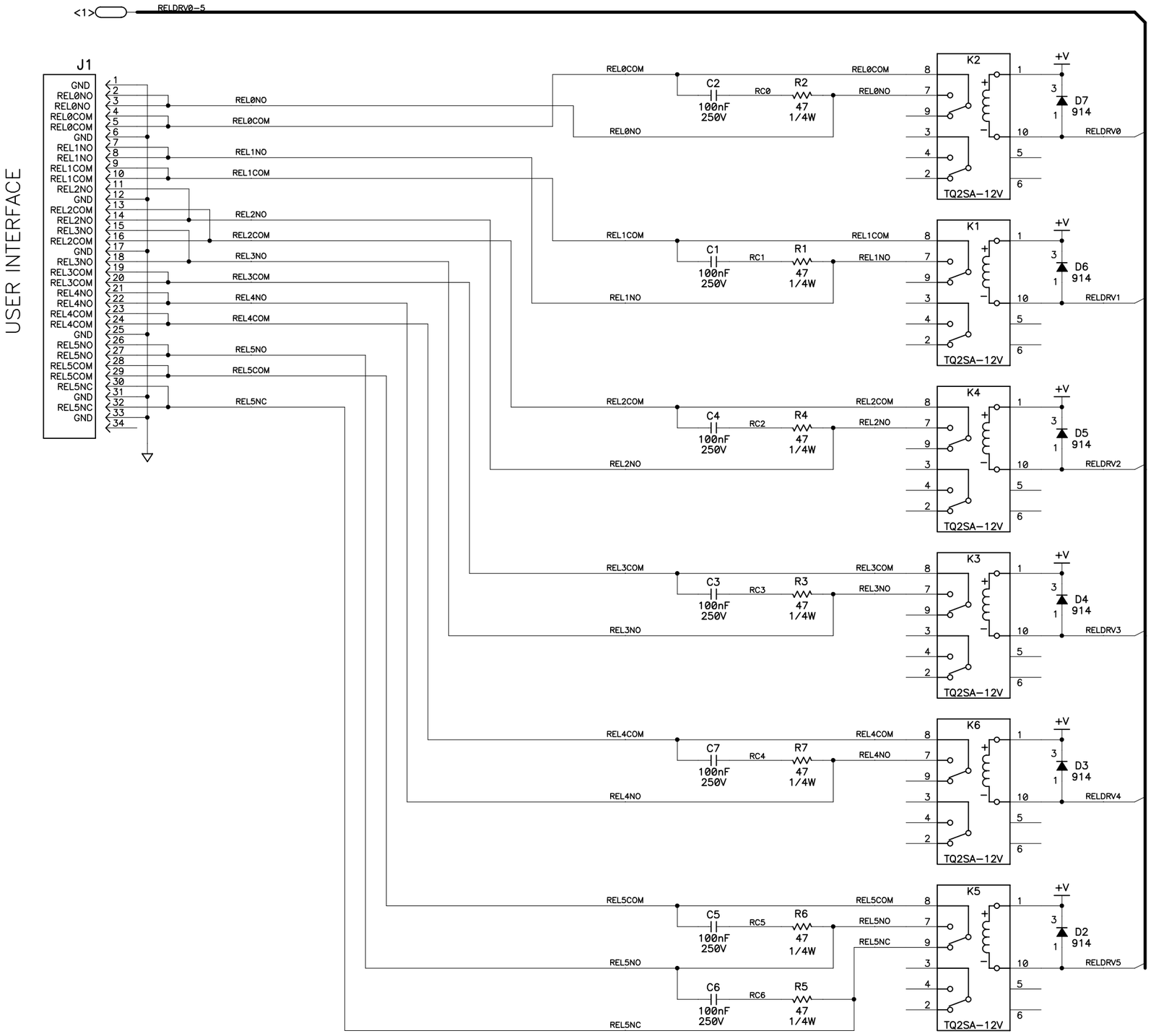
TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION					DEVICE: FILTER CAP REF DES(s)
		AGND	GND	+5V	+V_USER	NO CONNECTS	
U1	7812						
U2	LM311		4		8		C10
U3	74AHC32		3	5			C11
U4	74HC259		8	16			C12

NOTES: UNLESS OTHERWISE SPECIFIED;
1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (V_{CC}), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY ($\underline{V_{CC}}$).

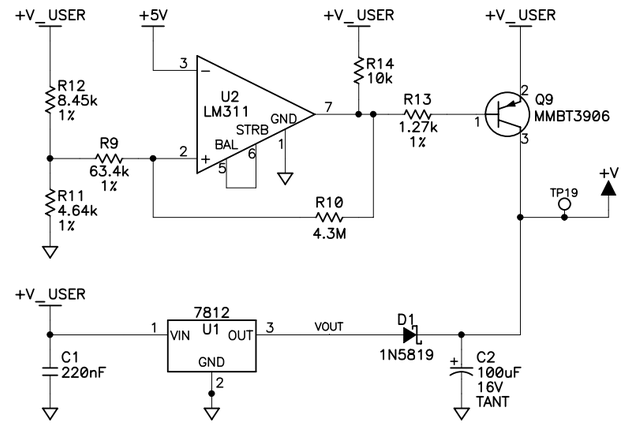
APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

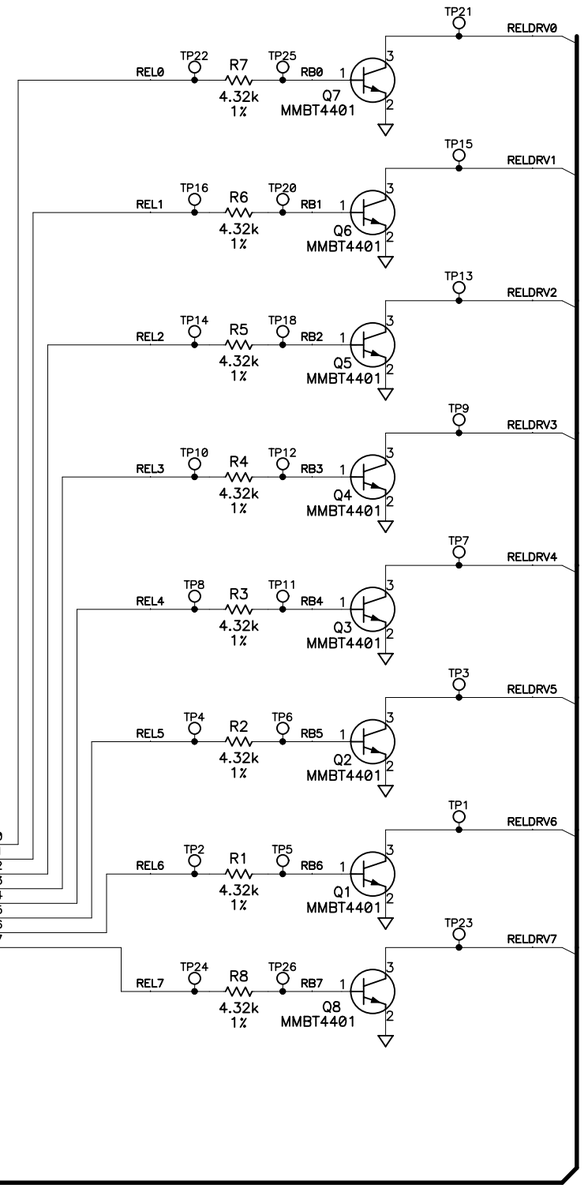
DRAWING CONTENT:		TITLE	
DRAWN BY: (INITIAL RELEASE) J.H.	26JAN00	SCHEMATIC DIAGRAM SR9500 SERIES RELAY BOARD	Z-WORLD 2900 SPAFFORD ST. DAVIS, CA 95616 530 - 757 - 4616
REVISED BY: K.SCHALLER	NOV1, 2001		
APPROVALS: INITIAL RELEASE			
PROJECT ENGINEER: DARREN MUSGROVE	11/20/00	SIZE B	DWG NO. 090-0098
ENGINEERING MANAGER: ____	____		
SIGNATURES	DATE	SCALE NONE	RELEASE DATE 11/20/00 SHEET 1 OF 2

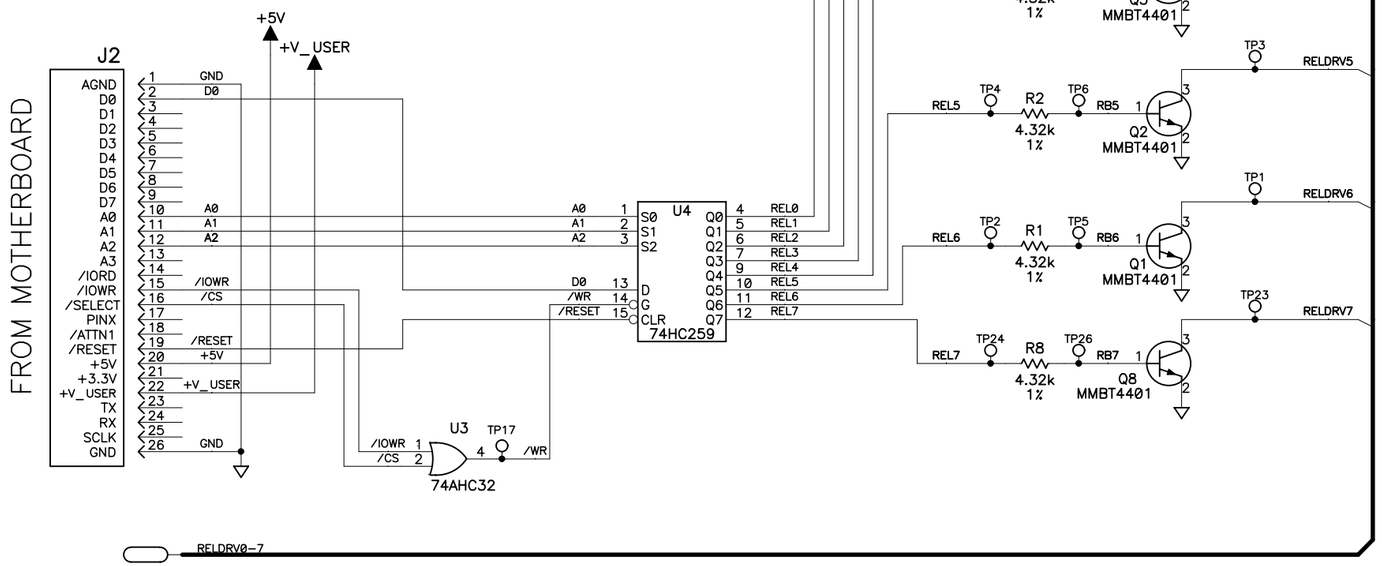


NOV1, 2001

SIZE	DWG NO.	
B	090-0098	
SCALE NONE	REV LTR	SHEET 2 OF 2



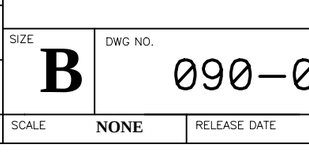




REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11266	INITIAL RELEASE A/W @ REV-A	XT	11/21/00	KIS	11/21/00
B	E11776	CORRECTED REFERENCE DESIGNATORS.	XT	2/15/02	KlS	2/15/02

TABLE A

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION					DEVICE: FILTER CAP REF DES(s)
		AGND	GND	+5V	+V_USER	NO CONNECTS	
U1	7812						
U2	LM311		4		8		C10
U3	74AHC32		3	5			C11
U4	74HC259		8	16			C12



NOTES: UNLESS OTHERWISE SPECIFIED;
1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) REP	19JUN00
REVISED BY: XUAN TRUONG	18DEC01
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	DATE

TITLE

SCHEMATIC DIAGRAM SR9510 RELAY 8 BOARD

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE **B** | DWG NO. 090-0108

SCALE **NONE** | RELEASE DATE | SHEET 1 OF 2

RELDRV0-7

USER INTERFACE

J1

GND	1	
REL0NO	2	REL0NO
REL0COM	3	REL0COM
REL0NC	4	REL0NC
REL1NO	5	REL1NO
GND	6	
REL1COM	7	REL1COM
REL1NC	8	REL1NC
REL2NO	9	REL2NO
REL2COM	10	REL2COM
REL2NC	11	REL2NC
GND	12	
REL3NO	13	REL3NO
REL3COM	14	REL3COM
REL3NC	15	REL3NC
REL4NO	16	REL4NO
GND	17	
REL4COM	18	REL4COM
REL4NC	19	REL4NC
REL5NO	20	REL5NO
REL5COM	21	REL5COM
REL5NC	22	REL5NC
REL6NO	23	REL6NO
REL6COM	24	REL6COM
GND	25	
REL6NC	26	REL6NC
REL7NO	27	REL7NO
REL7COM	28	REL7COM
REL7NC	29	REL7NC
	30	
GND	31	
	32	
	33	
	34	

K1 — TQ2SA-12V — D8 914 — +V — REL0COM, REL0NO, REL0NC → RELDRV0 → RELDRV0

K4 — TQ2SA-12V — D7 914 — +V — REL1COM, REL1NO, REL1NC → RELDRV1

K3 — TQ2SA-12V — D6 914 — +V — REL2COM, REL2NO, REL2NC → RELDRV2 → RELDRV2

K6 — TQ2SA-12V — D5 914 — +V — REL3COM, REL3NO, REL3NC → RELDRV3

K5 — TQ2SA-12V — D4 914 — +V — REL4COM, REL4NO, REL4NC → RELDRV4 → RELDRV4

K8 — TQ2SA-12V — D3 914 — +V — REL5COM, REL5NO, REL5NC → RELDRV5

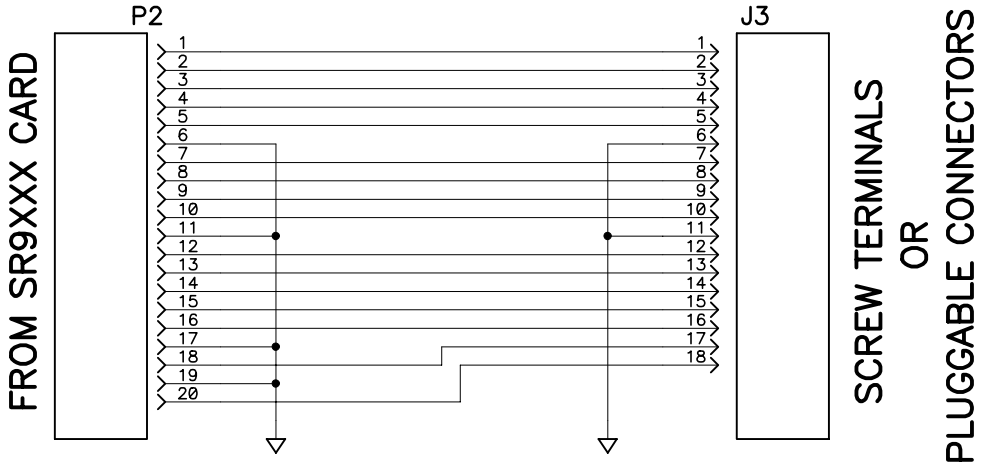
K7 — TQ2SA-12V — D2 914 — +V — REL6COM, REL6NO, REL6NC → RELDRV6 → RELDRV6

K2 — TQ2SA-12V — D9 914 — +V — REL7COM, REL7NO, REL7NC → RELDRV7

18DEC01

SIZE	DWG NO.	
B	090-0108	
SCALE NONE	REV LTR B	SHEET 2 OF 2

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11217	INITIAL RELEASE	XT	2/15/02	KlS	2/15/02
B	E11450	CONNECTED P2 PIN1 TO J3 PIN1, REMOVED GND	DM	15MAR01	KlS	3/20/00



APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) KEITH HOEK	08DEC99
REVISED BY: DARREN MUSGROVE	15MAR01
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	**DATE**

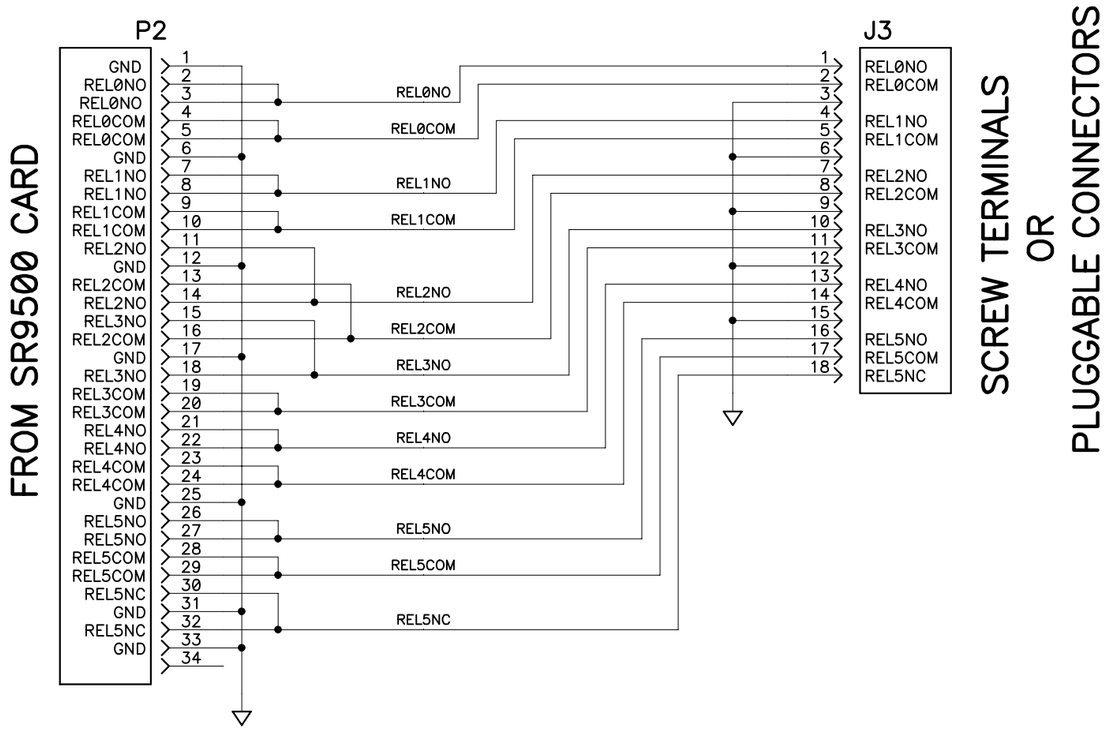
TITLE SCHEMATIC DIAGRAM
SR9XXX
18 POSITION
FIELD WIRING TERMINAL

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE A | DWG NO. 090–0102

SCALE NONE | RELEASE DATE | SHEET 1 OF 1

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11217	INITIAL RELEASE	XT	2/15/02	KlS	2/15/02



APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) KEITH HOEK	24MAY00
REVISED BY: KEITH HOEK	26SEP00
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	**DATE**

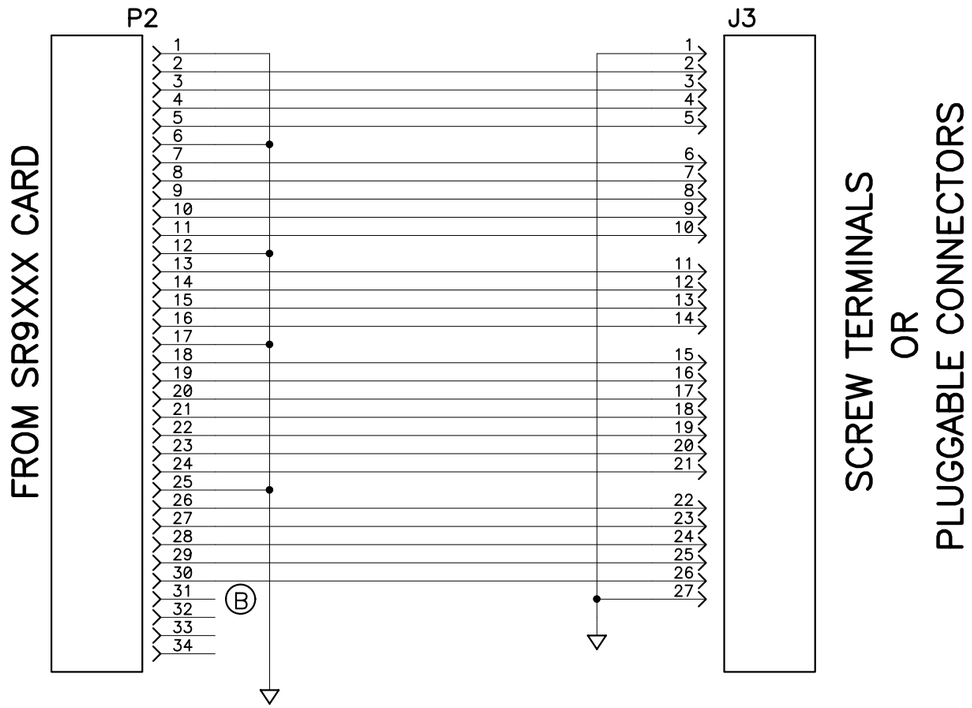
TITLE SCHEMATIC DIAGRAM
SR9500 SERIES
ONLY 18 POSITION
FIELD WIRING TERMINAL

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

SIZE A | DWG NO. 090–0106

SCALE NONE | RELEASE DATE | SHEET 1 OF 1

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11217	INITIAL RELEASE	XT	2/15/02	KlS	2/15/02
B	E11326	DISCONNECT PIN 31 FROM GND	DM	22DEC00	KlS	12/22/00



APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) KEITH HOEK	02DEC99
REVISED BY: ERIC PEAK	22DEC00
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: Darren Musgrove	7/20/01
ENGINEERING MANAGER: R.Matthews	2/15/02
SIGNATURES	**DATE**

TITLE SCHEMATIC DIAGRAM
SR9XXX SERIES
27 POSITION
FIELD WIRING TERMINAL

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757 - 4616

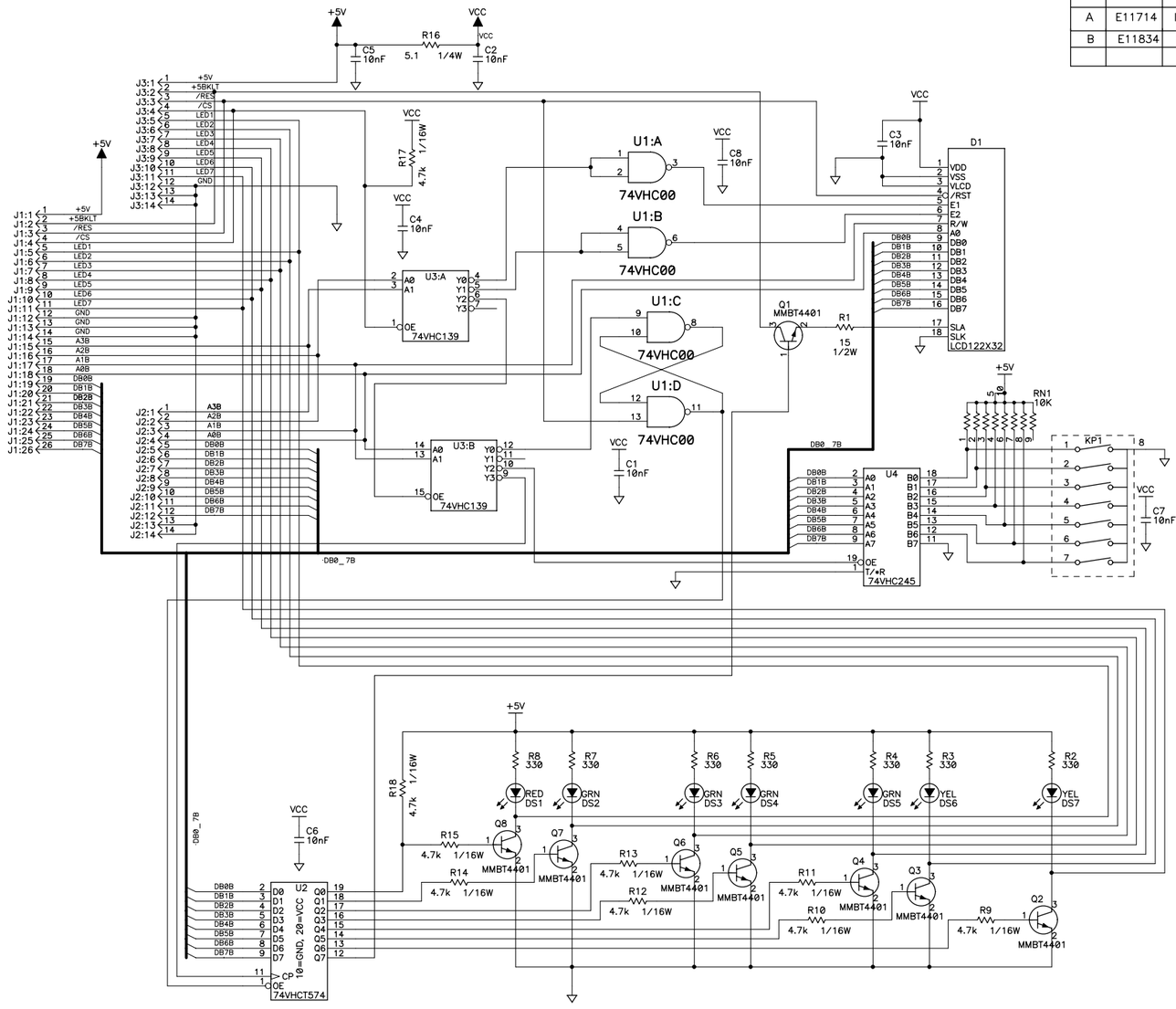
SIZE A | DWG NO. 090–0103

SCALE NONE | RELEASE DATE | SHEET 1 OF 1

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION OF CHANGE	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11714	INITIAL RELEASE	JF	11/1/01	KIS	11/1/01
B	E11834	RENAME SIGNAL LINE TO /CS FOR CONVENTION, CHANGE TITLE BLOCK	XT	2/15/02	KlS	2/15/02

POWER TABLE

REF DES	DEVICE	DEVICE VOLTAGE INFORMATION			DEVICE: FILTER CAP
		GND	VCC	NO CONNECTS	REF DES(s)
U1	74VHC00	7	14		C8
U2	74VHCT574	1, 8	16	12	C6
U3	74VHC139	8	16	10	C4
U4	74VHC245	1, 8, 11	16		C7



NOTES: UNLESS OTHERWISE SPECIFIED;
1. ALL RESISTOR VALUES ARE IN OHMS, 1/16W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:	

DRAWING CONTENT:	
DRAWN BY: (INITIAL RELEASE) J. FENG	23JUL01
REVISED BY: KRISTI SCHALLER	02/12/02
APPROVALS: INITIAL RELEASE	
PROJECT ENGINEER: J. FENG	10OCT01
ENGINEERING MANAGER: R.MATTHEWS	11/1/01
SIGNATURES	**DATE**

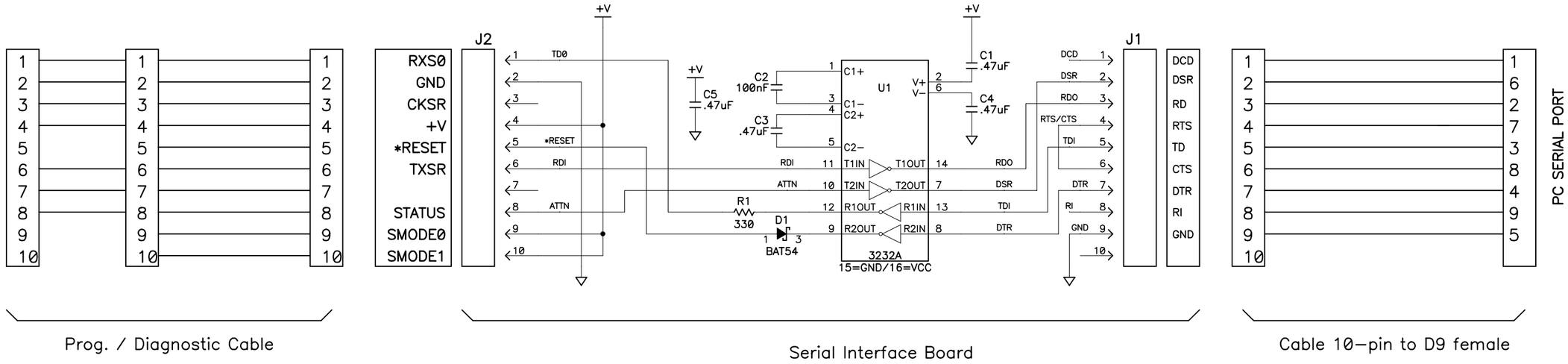
TITLE: SCHEMATIC DIAGRAM KEYPAD DISPLAY MODULE

Z-WORLD
2900 SPAFFORD ST.
DAVIS, CA 95616
530 - 757-4616

SIZE: C | DWG NO.: 090–0125

SCALE: NONE | RELEASE DATE: 2/15/02 | SHEET 1 OF 1

REVISION HISTORY			REVISION APPROVAL			
REV	ECO	DESCRIPTION	PROJECT ENGINEER	APPROVAL DATE	DOCUMENT CONTROL	APPROVAL DATE
A	E11523	INITIAL RELEASE OF SCHEMATIC	EP	5/14/01	KIS	5/14/01
B	E11691	CORRECT DE9 PINOUT	EP	10/5/01	KIS	10/4/01
C	E11816	ADD 3.3V CAPABILITY AND RED TUBING				



NOTES: UNLESS OTHERWISE SPECIFIED;
1. ALL RESISTOR VALUES ARE IN OHMS, 1/10W, 5%
2. ALL CAPACITORS ARE 50VDC OR HIGHER.
3. THE ORIGINATION SOURCE OF A VOLTAGE IS REPRESENTED BY (VCC), AND ALL REFERENCES TO THAT VOLTAGE ARE REPRESENTED BY (VCC).

APPEND THE FOLLOWING DOCUMENTS WHEN CHANGING THIS DOCUMENT:		DRAWING CONTENT:		TITLE	
		DRAWN BY: (INITIAL RELEASE) E. PEAK	14MAY01	SCHEMATIC DIAGRAM RABBIT PROG. CABLE 3.3V – 5.0V	2900 SPAFFORD ST. DAVIS, CA 95616 530 - 757 - 4616
		REVISED BY: D MUSGROVE	01/21/02		
		APPROVALS: INITIAL RELEASE			
		PROJECT ENGINEER: ERIC PEAK	5/14/01	SIZE B	DWG NO. 090–0128
		ENGINEERING MANAGER: R MATTHEWS	5/14/01		
		SIGNATURES	DATE	SCALE NONE	RELEASE DATE / SHEET 1 OF 1