PRODUCT MANUAL



RabbitCore RCM2200

C-Programmable Module with Ethernet

User's Manual

019–0097 • 090417–G

RabbitCore RCM2200 User's Manual

Part Number 019-0097 • 090417–G • Printed in U.S.A.

©2001–2009 Digi International Inc. • All rights reserved.

No part of the contents of this manual may be reproduced or transmitted in any form or by any means without the express written permission of Digi International.

Permission is granted to make one or more copies as long as the copyright page contained therein is included. These copies of the manuals may not be let or sold for any reason without the express written permission of Digi International.

Digi International reserves the right to make changes and improvements to its products without providing notice.

Trademarks

Rabbit and Dynamic C are registered trademarks of Digi International Inc.

Rabbit 2000 and RabbitCore are trademarks of Digi International Inc.

The latest revision of this manual is available on the Rabbit Web site, www.rabbit.com, for free, unregistered download.

Digi International Inc.

www.rabbit.com

TABLE OF CONTENTS

Chapter 1. Introduction 1

1.1 RCM2200 Features 1
1.2 Advantages of the RCM2200 2
1.3 Development and Evaluation Tools 3
1.3.1 Development Software 3
1.4 Development Kit Contents 3
1.5 How to Use This Manual 4
1.5.1 Additional Product Information 4
1.5.2 Online Documentation 4

Chapter 2. Getting Started 5

2.1 Connections 6
2.1.1 Attach Module to Prototyping Board 6
2.1.2 Connect Programming Cable 7
2.1.3 Connect Power 8
2.1.4 Alternate Power Supply Connections 9
2.2 Run a Sample Program 9
2.2.1 Troubleshooting 9
2.3 Where Do I Go From Here? 10
2.3.1 Technical Support 10

Chapter 3. Running Sample Programs 11

3.1 Sample Programs 11
3.1.1 Getting to Know the RCM2200 12
3.1.2 Serial Communication 14
3.1.3 Other Sample Programs 15
3.1.4 Sample Program Descriptions 16
3.1.4.1 FLASHLED.C 16
3.1.4.2 FLASHLEDS.C 17
3.1.4.3 TOGGLELED.C 18

Chapter 4. Hardware Reference 19

4.1 RCM2200 Digital Inputs and Outputs 19
4.1.1 Dedicated Inputs 20
4.1.2 Dedicated Outputs 20
4.1.3 Memory I/O Interface 20
4.1.4 Other Inputs and Outputs 22
4.2 Serial Communication 23
4.2.1 Serial Ports 23
4.2.2 Ethernet Port 23
4.2.3 Programming Port 24
4.3 Serial Programming Cable 25
4.3.1 Changing Between Program Mode and Run Mode 25
4.3.2 Standalone Operation of the RCM2200 26

4.4 Memory 27
4.4.1 SRAM 27
4.4.2 Flash EPROM 27
4.4.3 Dynamic C BIOS Source Files 27
4.5 Other Hardware 28
4.5.1 Clock Doubler 28
4.5.2 Spectrum Spreader 29

Chapter 5. Software Reference 31
5.1 More About Dynamic C 31
5.2 I/O 33
5.2.1 PCLK Output 33
5.2.2 External Interrupts 33
5.3 Serial Communication Drivers 34
5.4 TCP/IP Drivers 34
5.5 Upgrading Dynamic C 35
5.5.1 Extras 35

Chapter 6. Using the TCP/IP Features 37
6.1 TCP/IP Connections 37
6.2 Running TCP/IP Sample Programs 39
6.3 IP Addresses Explained 41
6.4 How IP Addresses are Used 42
6.5 Dynamically Assigned Internet Addresses 43
6.6 Placing Your Device on the Internet 44
6.7 How to Set IP Addresses in the Sample Programs 45
6.8 How to Set Up Your Computer for Direct Connect 46
6.9 Run the PINGME.C Sample Programs 47
6.10 Running More Sample Programs With Direct Connect 47
6.11 Where Do I Go From Here? 48

Appendix A. RabbitCore RCM2200 Specifications 49
A.1 Electrical and Mechanical Characteristics 50
A.1.1 Headers 53
A.1.2 Physical Mounting 53
A.2 Bus Loading 54
A.3 Rabbit 2000 DC Characteristics 56
A.4 I/O Buffer Sourcing and Sinking Limit 57
A.5 Jumper Configurations 58
A.6 Conformal Coating 59

Appendix B. Prototyping Board 61
B.1 Prototyping Board 62
B.1.1 Prototyping Board Features 63
B.1.2 Prototyping Board Expansion 64
B.2 Mechanical Dimensions and Layout 65
B.3 Power Supply 66
B.4 Using the Prototyping Board 66
B.4.1 Adding Other Components 69

Appendix C. Power Supply 71
C.1 Power Supplies 71
C.1.1 Battery-Backup Circuits 71
C.1.2 Reset Generator 72
C.2 Chip Select Circuit 73

Appendix D. Sample Circuits **75**
D.1 RS-232/RS-485 Serial Communication 76
D.2 Keypad and LCD Connections 77
D.3 External Memory 78
D.4 D/A Converter 79

Index **81**

Schematics **83**

1. INTRODUCTION

The RCM2200 RabbitCore module is designed to be the heart of embedded control systems. The RCM2200 features an integrated Ethernet port and provides for LAN and Internet-enabled systems to be built as easily as serial-communication systems.

Throughout this manual, the term RCM2200 refers to the complete series of RCM2200 RabbitCore modules unless other production models are referred to specifically.

The RCM2200 has a Rabbit 2000 microprocessor operating at 22.1 MHz, static RAM, flash memory, two clocks (main oscillator and timekeeping), and the circuitry necessary for reset and management of battery backup of the Rabbit 2000's internal real-time clock and the static RAM. Two 26-pin headers bring out the Rabbit 2000 I/O bus lines, address lines, data lines, parallel ports, and serial ports.

The RCM2200 receives its +5 V power from the user board on which it is mounted. The RabbitCore RCM2200 can interface with all kinds of CMOS-compatible digital devices through the user board.

1.1 RCM2200 Features

- Small size: 1.60" × 2.30" × 0.86"
  (41 mm × 58 mm × 22 mm)
- Microprocessor: Rabbit 2000 running at 22.1 MHz
- 26 parallel I/O lines: 16 configurable for input or output, 7 fixed inputs, 3 fixed outputs
- 8 data lines (D0–D7)
- 4 address lines (A0–A3)
- Memory I/0 read, write
- External reset input
- Five 8-bit timers (cascadable in pairs) and two 10-bit timers
- 256K–512K flash memory, 128K–512K SRAM
- Real-time clock
- Watchdog supervisor

- Provision for customer-supplied backup battery via connections on header J5
- 10/100-compatible RJ-45 Ethernet port with 10Base-T interface (Ethernet jack not installed on all models)
- Raw Ethernet and two associated LED control signals available on 26-pin header
- Three CMOS-compatible serial ports: maximum asynchronous baud rate of 691,200 bps, maximum synchronous baud rate of 5,529,600 bps. One port is configurable as a clocked port.
- Six additional I/O lines are located on the programming port, can be used as I/O lines when the programming port is not being used for programming or in-circuit debugging—one synchronous serial port can also be used as two general CMOS inputs and one general CMOS output, and there are two additional inputs and one additional output.

Appendix A, "RabbitCore RCM2200 Specifications," provides detailed specifications for the RCM2200.

In addition, three different RCM2200 models are available. A variant of the RCM2200, the RCM2300, omits the Ethernet connectivity but offers a much smaller footprint, one-half the size of the RCM2200.

1.2 Advantages of the RCM2200

- Fast time to market using a fully engineered, "ready to run" microprocessor core.
- Competitive pricing when compared with the alternative of purchasing and assembling individual components.
- Easy C-language program development and debugging, including rapid production loading of programs.
- Generous memory size allows large programs with tens of thousands of lines of code, and substantial data storage.
- Integrated Ethernet port for network connectivity, royalty-free TCP/IP software.

1.3 Development and Evaluation Tools

A complete Development Kit, including a Prototyping Board and Dynamic C development software, is available for the RCM2200. The Development Kit puts together the essentials you need to design an embedded microprocessor-based system rapidly and efficiently.

1.3.1 Development Software

The RCM2200 module uses the Dynamic C development environment for rapid creation and debugging of runtime applications. Dynamic C provides a complete development environment with integrated editor, compiler and source-level debugger. It interfaces directly with the target system, eliminating the need for complex and unreliable in-circuit emulators.

> **NOTE:** The RCM2200 module requires Dynamic C v7.04 or later for development. A compatible version is included on the Development Kit CD-ROM.

1.4 Development Kit Contents

The RCM2200 Development Kit contains the following items:

- RCM2200 module with 10Base-T Ethernet port, 256K flash memory, and 128K SRAM.
- RCM2200/RCM2300 Prototyping Board.
- Wall transformer power supply, 12 V DC, 1 A. (Included only with Development Kits sold for the North American market. Overseas users will have to substitute a power supply compatible with local mains power.)
- 10-pin header to DB9 programming cable with integrated level-matching circuitry.
- ***Dynamic C*** CD-ROM, with complete product documentation on disk.
- ***Getting Started*** instructions.
- ***Rabbit 2000 Processor Easy Reference*** poster.
- Registration card.

1.5 How to Use This Manual

This user's manual is intended to give users detailed information on the RCM2200 module. It does not contain detailed information on the Dynamic C development environment or the TCP/IP software support for the integrated Ethernet port. Most users will want more detailed information on some or all of these topics in order to put the RCM2200 module to effective use.

1.5.1 Additional Product Information

In addition to the product-specific information contained in the ***RabbitCore RCM2200 User's Manual*** (this manual), several higher level reference manuals are provided in HTML and PDF form on the accompanying CD-ROM. Advanced users will find these references valuable in developing systems based on the RCM2200 modules:

- ***Dynamic C User's Manual***
- ***An Introduction to TCP/IP***
- ***Dynamic C TCP/IP User's Manual***
- ***Rabbit 2000 Microprocessor User's Manual***

1.5.2 Online Documentation

The online documentation is installed along with Dynamic C, and an icon for the documentation menu is placed on the workstation's desktop. Double-click this icon to reach the menu. If the icon is missing, use your browser to find and load **default.htm** in the **docs** folder, found in the Dynamic C installation folder.

The latest versions of all documents are always available for free, unregistered download from our Web sites as well.

2. GETTING STARTED

This chapter describes the RCM2200 hardware in more detail, and explains how to set up and use the accompanying Prototyping Board.

NOTE: This chapter (and this manual) assume that you have the RCM2200 Development Kit. If you purchased an RCM2200 module by itself, you will have to adapt the information in this chapter and elsewhere to your test and development setup.

2.1 Connections

There are four steps to connecting the Prototyping Board for use with Dynamic C and the sample programs:

1. Attach the RCM2200 module to the Prototyping Board.
2. Connect the programming cable between the RCM2200 module and the workstation PC.
3. Connect the power supply to the Prototyping Board.

2.1.1 Attach Module to Prototyping Board

Turn the RCM2200 module so that the Ethernet connector end of the module extends off the Prototyping Board, as shown in Figure 1 below. Align the pins from headers J4 and J5 on the bottom side of the RCM2200 with header sockets J1 and J2 on the Prototyping Board.

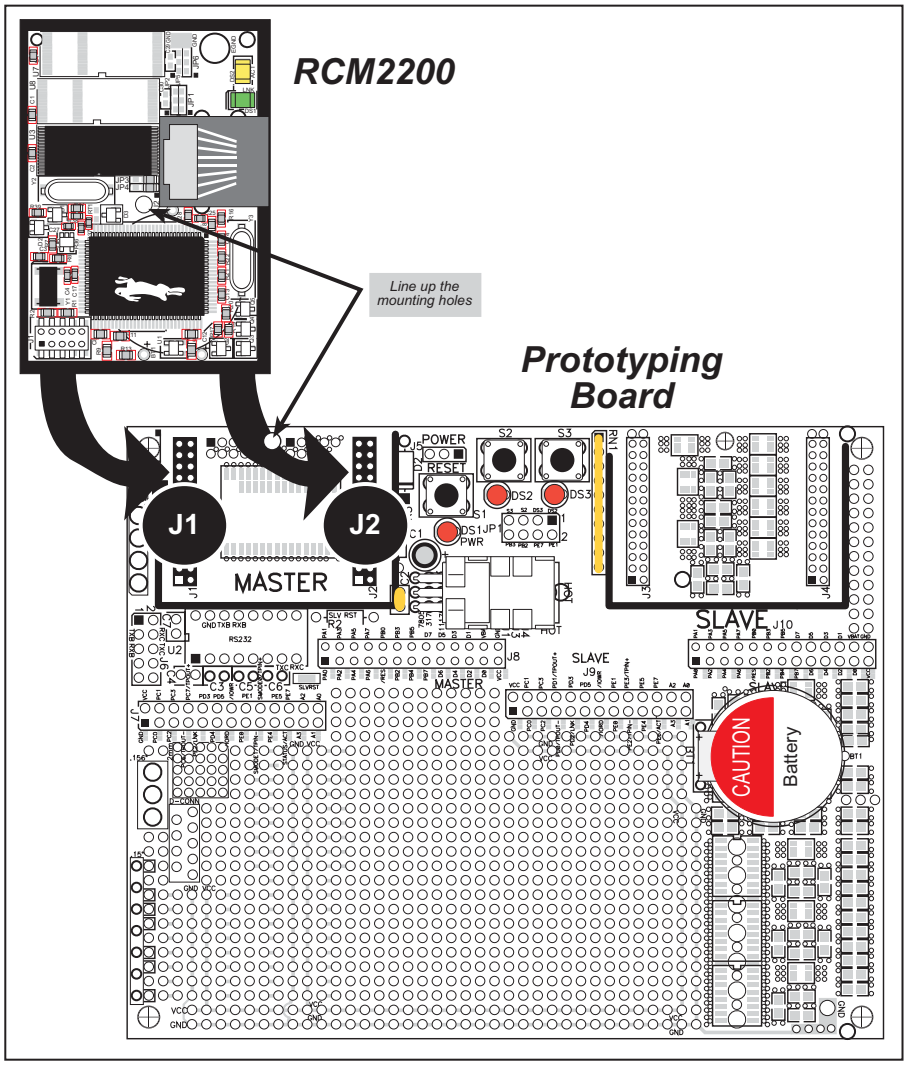


Figure 1. Installing the RCM2200 on the Prototyping Board

Although you can install a single module into either the **MASTER** or the **SLAVE** position on the Prototyping Board, all the Prototyping Board features (switches, LEDs, serial port drivers, etc.) are connected to the **MASTER** position. We recommend you install the module in the **MASTER** position.

NOTE: It is important that you line up the pins on headers J4 and J5 of the RCM2200 exactly with the corresponding pins of header sockets J1 and J2 on the Prototyping Board. The header pins may become bent or damaged if the pin alignment is offset, and the module will not work. Permanent electrical damage to the module may also result if a misaligned module is powered up.

Press the module's pins firmly into the Prototyping Board header sockets.

2.1.2 Connect Programming Cable

The programming cable connects the RCM2200 module to the PC running Dynamic C to download programs and to monitor the RCM2200 for debugging.

Connect the 10-pin connector of the programming cable labeled **PROG** to header J1 on the RCM2200 module as shown in Figure 2. Be sure to orient the marked (usually red) edge of the cable towards pin 1 of the connector. (Do not use the **DIAG** connector, which is used for a normal serial connection.)

Connect the other end of the programming cable to a COM port on your PC. Make a note of the port to which you connect the cable, as Dynamic C needs to have this parameter configured when it is installed.

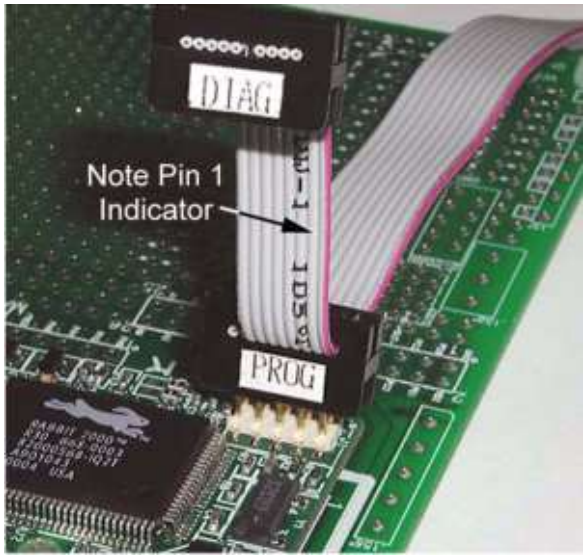


Figure 2. Connect Programming Cable to RCM2200

NOTE: COM 1 is the default port used by Dynamic C.

NOTE: Some PCs now come equipped only with a USB port. It may be possible to use an RS-232/USB converter (Part No. 20-151-0178) with the programming cable supplied with the RCM2200 Development Kit. Note that not all RS-232/USB converters work with Dynamic C.

2.1.3 Connect Power

When all other connections have been made, you can connect power to the RCM2200 Prototyping Board.

First, prepare the AC adapter for the country where it will be used by selecting the plug. The RCM2200 Development Kit presently includes Canada/Japan/U.S., Australia/N.Z., U.K., and European style plugs. Snap in the top of the plug assembly into the slot at the top of the AC adapter as shown in Figure 3, then press down on the spring-loaded clip below the plug assembly to allow the plug assembly to click into place.

Connect the AC adapter to 3-pin header J5 on the Prototyping Board as shown in Figure 3 below. The connector may be attached either way as long as it is not offset to one side.

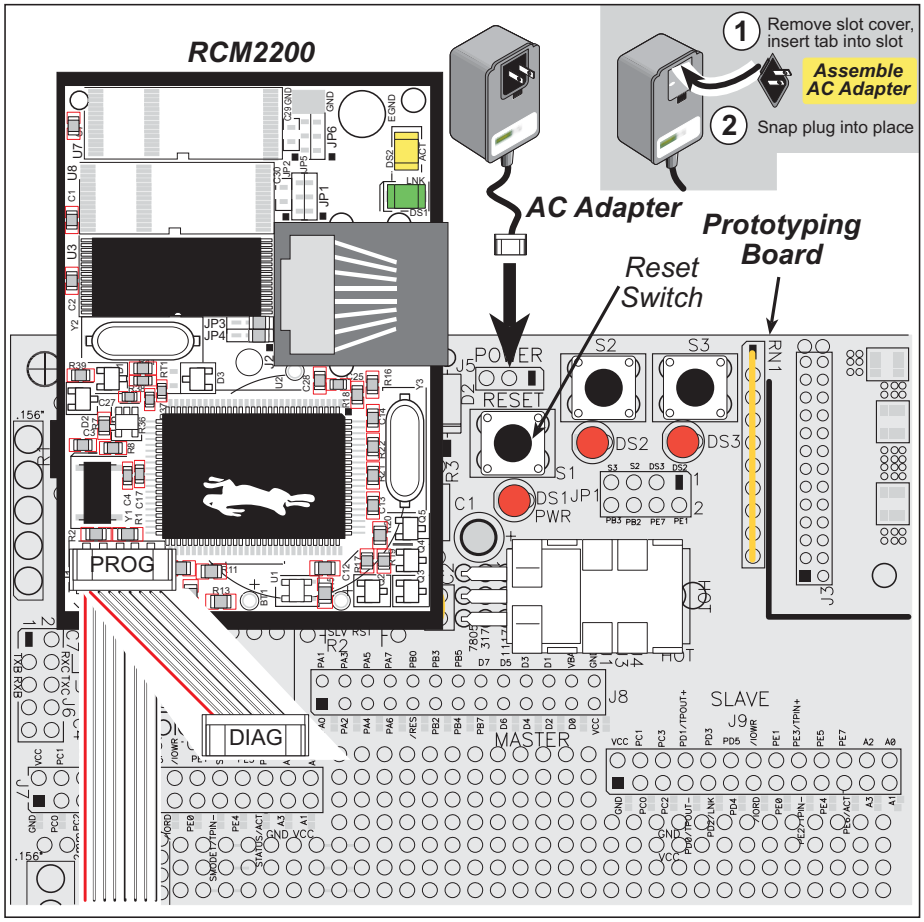


Figure 3. Power Supply Connections

Plug in the AC adapter. The power LED on the Prototyping Board should light up. The RCM2200 and the Prototyping Board are now ready to be used.

NOTE: A **RESET** button is provided on the Prototyping Board to allow hardware reset without disconnecting power.

To power down the Prototyping Board, unplug the power connector from J5. You should disconnect power before making any circuit adjustments in the prototyping area, changing any connections to the board, or removing the RabbitCore module from the board.

2.1.4 Alternate Power Supply Connections

Development kits sold outside North America before 2009 included a header connector that could be connected to 3-pin header J5 on the Prototyping Board. The red and black wires from the connector could then be connected to the positive and negative connections on your power supply. The power supply should deliver 8 V–24 V DC at 8 W.

2.2 Run a Sample Program

Once the RCM2200 is connected as described in the preceding pages, start Dynamic C by double-clicking on the Dynamic C icon on your desktop or in your **Start** menu. Dynamic C uses the serial port specified during installation.

If you are using a USB port to connect your computer to the RCM2200 module, choose **Options > Project Options** and select "Use USB to Serial Converter" under the **Communications** tab, then click **OK**.

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), then compile and run it by pressing **F9** or by selecting **Run** in the **Run** menu. The **STDIO** window will open and will display a small square bouncing around in a box.

2.2.1 Troubleshooting

If Dynamic C cannot find the target system (error message **"No Rabbit Processor Detected."**):

- Check that the RCM2200 is powered correctly — the red power LED on the Prototyping Board should be lit when the RCM2200 is mounted on the Prototyping Board and the AC adapter is plugged in.
- Check both ends of the programming cable to ensure that they are firmly plugged into the PC and the **PROG** connector, not the **DIAG** connector, is plugged in to the programming port on the RCM2200 with the marked (colored) edge of the programming cable towards pin 1 of the programming header.
- Ensure that the RCM2200 module is firmly and correctly installed in its connectors on the Prototyping Board.
- Dynamic C uses the COM port specified during installation. Select a different COM port within Dynamic C. From the **Options** menu, select **Project Options**, then select **Communications**. Select another COM port from the list, then click OK. Press **<Ctrl-Y>** to force Dynamic C to recompile the BIOS. If Dynamic C still reports it is unable to locate the target system, repeat the above steps until you locate the COM port used by the programming cable.

If Dynamic C appears to compile the BIOS successfully, but you then receive a communication error message when you compile and load the sample program, it is possible that your PC cannot handle the higher program-loading baud rate. Try changing the maximum download rate to a slower baud rate as follows.

- Locate the **Serial Options** dialog in the Dynamic C **Options > Project Options > Communications** menu. Select a slower Max download baud rate.

If a program compiles and loads, but then loses target communication before you can begin debugging, it is possible that your PC cannot handle the default debugging baud rate. Try lowering the debugging baud rate as follows.

- Locate the **Serial Options** dialog in the Dynamic C **Options > Project Options > Communications** menu. Choose a lower debug baud rate.

2.3 Where Do I Go From Here?

If everything appears to be working, we recommend the following sequence of action:

1. Run all of the sample programs described in Chapter 3 to get a basic familiarity with Dynamic C and the RCM2200 module's capabilities.
2. For further development, refer to the ***RabbitCore RCM2200 User's Manual*** for details of the module's hardware and software components.

   A documentation icon should have been installed on your workstation's desktop; click on it to reach the documentation menu. You can create a new desktop icon that points to **default.htm** in the **docs** folder in the Dynamic C installation folder.
3. For advanced development topics, refer to the ***Dynamic C User's Manual*** and the ***Dynamic C TCP/IP User's Manual***, also in the online documentation set.

2.3.1 Technical Support

NOTE: If you purchased your RCM2200 through a distributor or through a Rabbit partner, contact the distributor or partner first for technical support.

If there are any problems at this point:

- Use the Dynamic C **Help** menu to get further assistance with Dynamic C.
- Check the Rabbit Technical Bulletin Board and forums at www.rabbit.com/support/bb/ and at www.rabbit.com/forums/.
- Use the Technical Support e-mail form at www.rabbit.com/support/.

3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM2200 (and for all other Rabbit hardware), you must install and use Dynamic C. This chapter provides a tour of the sample programs for the RCM2200.

3.1 Sample Programs

To help familiarize you with the RCM2200 modules, several sample Dynamic C programs have been included. Loading, executing and studying these programs will give you a solid hands-on overview of the RC M2200's capabilities, as well as a quick start with Dynamic C as an application development tool. These programs are intended to serve as tutorials, but then can also be used as starting points or building blocks for your own applications.

> **NOTE:** It is assumed in this section that you have at least an elementary grasp of ANSI C. If you do not, see the introductory pages of the ***Dynamic C User's Manual*** for a suggested reading list.

Each sample program has comments that describe the purpose and function of the program.

Before running any of these sample program, make sure that your RCM2200 is connected to the Prototyping Board and to your PC as described in Section 2.1, "Connections." To run a sample program, open it with the **File** menu (if it is not already open), then compile and run it by pressing **F9** or by selecting **Run** in the **Run** menu.

Sample programs are provided in the Dynamic C **SAMPLES** folder. Two folders contain sample programs that illustrate features unique to the RCM2200.

- **RCM2200**—Demonstrates the basic operation and the Ethernet functionality of the RCM2200.
- **TCPIP**—Demonstrates more advanced TCP/IP programming for Rabbit's Ethernet-enabled Rabbit-based boards.

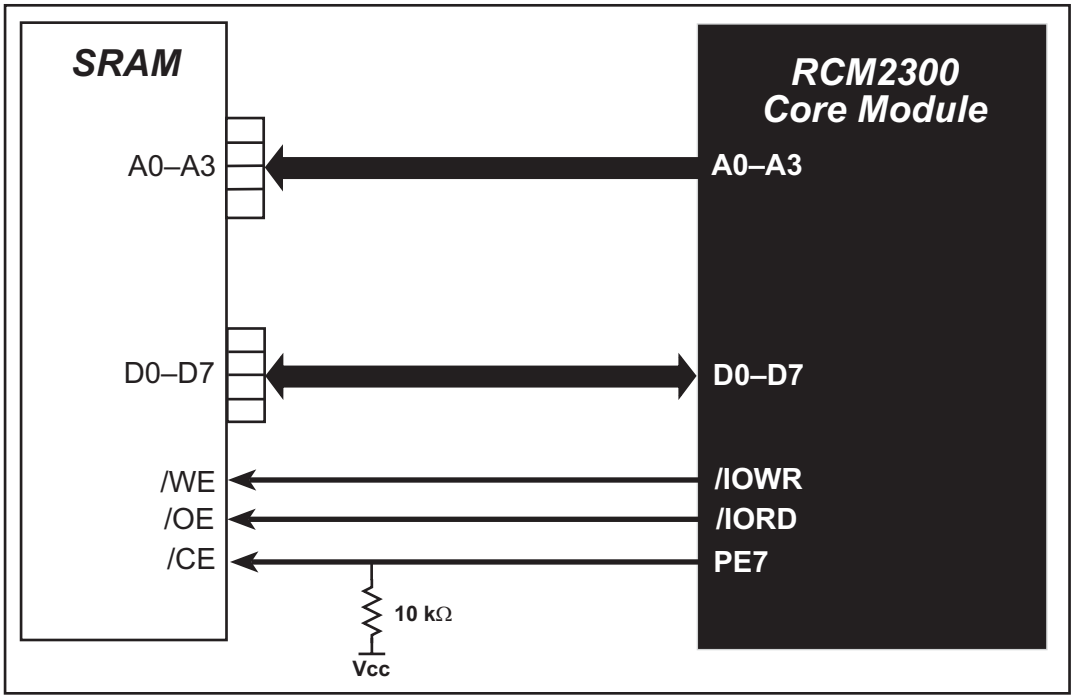
Complete information on Dynamic C is provided in the ***Dynamic C User's Manual***.

3.1.1 Getting to Know the RCM2200

The following sample programs can be found in the **SAMPLES\RCM2200** folder.

- **EXTSRAM.C**—demonstrates the setup and simple addressing to an external SRAM. This program first maps the external SRAM to the I/O Bank 7 register with a maximum of 15 wait states, chip select strobe (PE7), and allows writes. The first 256 bytes of SRAM are cleared and read back. Values are then written to the same area and are read back. The Dynamic C **STDIO** window will indicate if writes and reads did not occur

  Connect an external SRAM as shown below before you run this sample program.

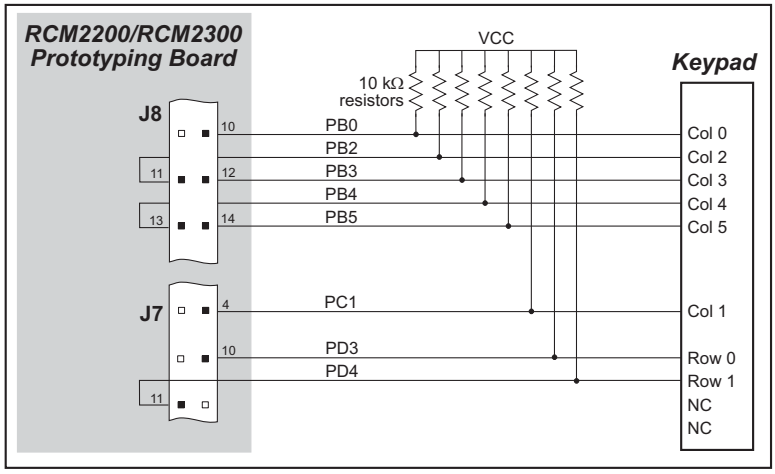


- **FLASHLED.C**—repeatedly flashes LED DS3 on the Prototyping Board on and off. LED DS3 is controlled by Parallel Port E bit 7 (PE7). LED DS2 will remain on continuously.
- **FLASHLEDS.C**—demonstrates the use of coding with assembly instructions, cofunctions, and costatements to flash LEDs DS2 and DS3 on the Prototyping Board on and off. LEDs DS2 and DS3 are controlled by Parallel Port E bit 1 (PE1) and Parallel Port E bit 7 (PE7).Once you have compiled this program and it is running, LEDs DS2 and DS3 will flash on/off at different rates.
- **TOGGLELED.C**—demonstrates the use of costatements to detect switch presses using the press-and-release method of debouncing. As soon as the sample program starts running, LED DS2 on the Prototyping Board (which is controlled by PE1) starts flashing once per second. Press switch S3 on the Prototyping Board (which is connected to PB3) to toggle LED DS3 on the Prototyping Board (which is controlled by PE7) on and off. The pushbutton switch is debounced by the software.

- **KEYLCD.C**—demonstrates a simple setup for a 2 × 6 keypad and a 2 × 20 LCD.

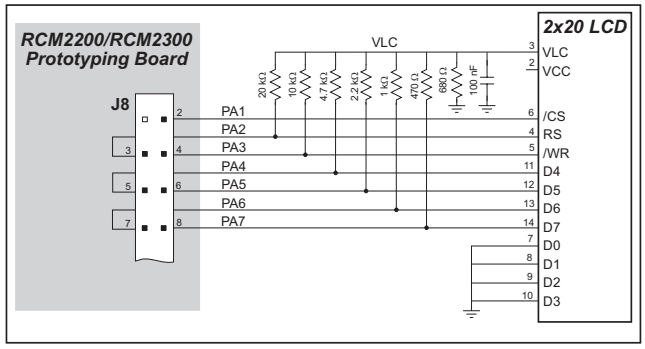
  Connect the keypad to Parallel Ports B, C, and D.

  PB0—Keypad Col 0
  PC1—Keypad Col 1
  PB2—Keypad Col 2
  PB3—Keypad Col 3
  PB4—Keypad Col 4
  PB5—Keypad Col 5
  PD3—Keypad Row 0
  PD4—Keypad Row 1



  Connect the LCD to Parallel Port A.

  PA0—backlight (if connected)
  PA1—LCD /CS
  PA2—LCD RS (High = Control, Low = Data) / LCD Contrast 0
  PA3—LCD /WR/ LCD Contrast 1
  PA4—LCD D4 / LCD Contrast 2
  PA5—LCD D5 / LCD Contrast 3
  PA6—LCD D6 / LCD Contrast 4
  PA7—LCD D7 / LCD Contrast 5

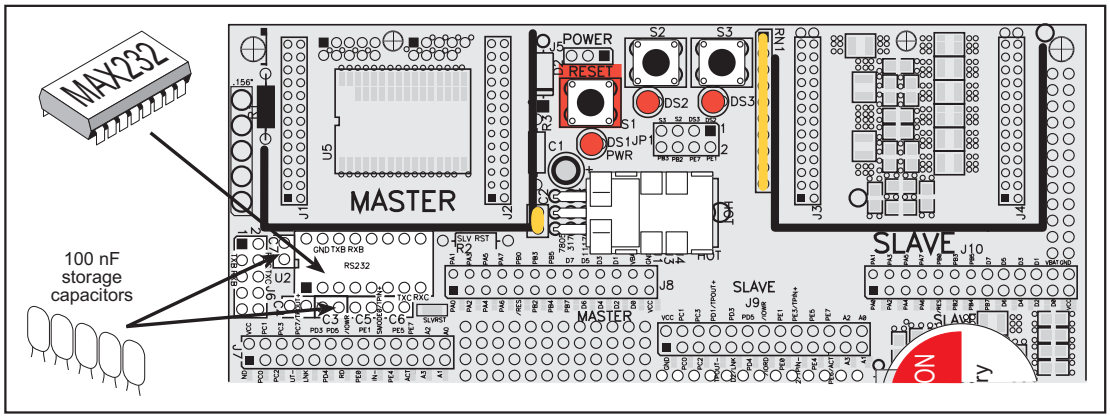


  Once the connections have been made and the sample program is running, the LCD will display two rows of 6 dots, each dot representing the corresponding key. When a key is pressed, the corresponding dot will become an asterisk.

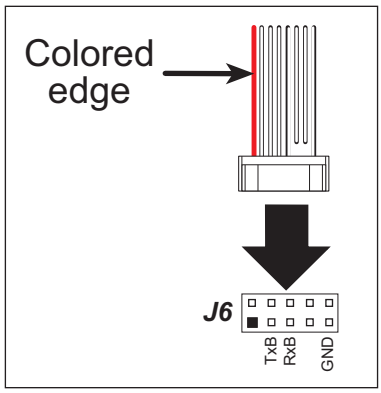
3.1.2 Serial Communication

The following sample programs can be found in the **SAMPLES\RCM2200** folder.

One sample programs, **PUTS.C** is available to illustrate RS-232 communication. To run this sample program, you will have to add an RS-232 transceiver such as the MAX232 at location U2 and five 100 nF capacitors at C3–C7 on the Prototyping Board. Also install a 2 × 5 IDC header with a pitch of 0.1" at J6 to interface the RS-232 signals. The diagram shows the connections.



Once the sample program is running, you may use a 10-pin header to DB9 cable (for example, Part No. 540-0009) to connect header J6 to your PC COM port (you will have to disconnect the programming cable from both the RCM2200 and the PC if you only have one PC COM port, then press the **RESET** button on the Prototyping Board). Line up the colored edge of the cable with pin 1 on header J6 as shown in the diagram (pin 1 is indicated by a small square on the Prototyping Board silkscreen).



This program writes a null terminated string over Serial Port B. Use a serial utility such as HyperTerminal or Tera Term to view the string. Use the following configuration for your serial utility.

Bits per second: 19200

Data bits: 8

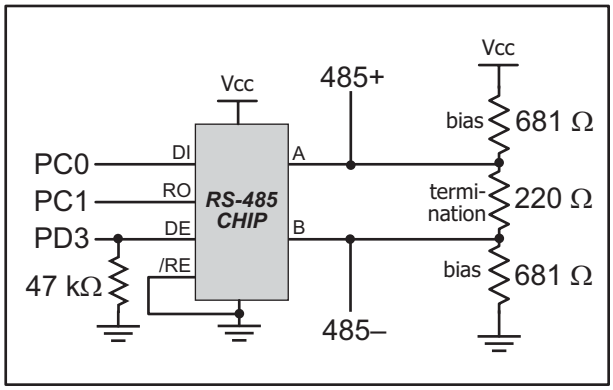
Parity: None

Stop bits: 1

Flow control: None

Two sample programs, **MASTER.C** and **SLAVE.C**, are available to illustrate RS-485 master/slave communication. To run these sample programs, you will need a second Rabbit-based system with RS-485, and you will also have to add an RS-485 transceiver such as the SP483E and bias resistors to the Prototyping Board.

The diagram shows the connections. You will have to connect PC0 and PC1 (Serial Port D) on the Prototyping Board to the RS-485 transceiver, and you will connect PD3 to the RS-485 transceiver to enable or disable the RS-485 transmitter.



The RS-485 connections between the slave and master devices are as follows.

- RS485+ to RS485+
- RS485– to RS485–
- GND to GND

- **MASTER.C**—This program demonstrates a simple RS-485 transmission of lower case letters to a slave RCM2200. The slave will send back converted upper case letters back to the master RCM2200 and display them in the **STDIO** window. Use **SLAVE.C** to program the slave RCM2200—reset the slave before you run **MASTER.C** on the master.
- **SLAVE.C**—This program demonstrates a simple RS-485 transmission of lower case letters to a master RCM2200. The slave will send back converted upper case letters back to the master RCM2200 and display them in the **STDIO** window. Compile and run this program on the slave before you use **MASTER.C** to program the master.

3.1.3 Other Sample Programs

Section 6.2 covers how to run the TCP/IP sample programs, which are then described in detail.

3.1.4 Sample Program Descriptions

3.1.4.1 FLASHLED.C

This program is about as simple as a Dynamic C application can get—the equivalent of the traditional "Hello, world!" program found in most basic programming tutorials. If you are familiar with ANSI C, you should have no trouble reading through the source code and understanding it.

The only new element in this sample application should be Dynamic C's handling of the Rabbit microprocessor's parallel ports. The program:

4. Initializes the pins of Port A as outputs.
5. Sets all of the pins of Port A high, turning off the attached LEDs.
6. Starts an endless loop with a **`for(;;)`** expression, and within that loop:
    - Writes a bit to turn bit 1 off, lighting LED DS3;
    - Waits through a delay loop;
    - Writes a bit to turn bit 1 on, turning off the LED;
    - Waits through a second delay loop;

    These steps repeat as long as the program is allowed to run.

You can change the flash rate of the LED by adjusting the loop values in the two **`for`** expressions. The first loop controls the LED's "off" time; the second loop controls its "on" time.

> **NOTE:** Since the variable **`j`** is defined as type **`int`**, the range for **`j`** must be between 0 and 32767. To permit larger values and thus longer delays, change the declaration of **`j`** to **`unsigned int`** or **`long`**.

More Information

See the section on primitive data types, and the entries for the library functions **`WrPortI( )`** and **`BitWrPortI( )`** in the ***Dynamic C User's Manual***.

3.1.4.2 FLASHLEDS.C

In addition to Dynamic C's implementation of C-language programming for embedded systems, it supports assembly-language programming for very efficient processor-level control of the module hardware and program flow. This application is similar to **FLASHLED.C** and **TOGGLELED.C**, but uses assembly language for the low-level port control within *cofunctions*, another powerful multitasking tool.

Dynamic C permits the use of assembly language statements within C code. This program creates three functions using assembly language statements, then creates a C cofunction to call two of them. That cofunction is then called within **main()**.

Within each of the C-like functions, the **#asm** and **#endasm** directives are used to indicate the beginning and end of the assembly language statements.

In the function **initialize_ports( )**, port A is initialized to be all outputs while bit 0 of port E is initialized to be an output.

In the function **ledon()**, a 0 is written to the port A bit corresponding to the desired LED (0, which equals DS3, or 1 which equals DS4), turning that LED on. The **ledoff( )** function works exactly the same way except that a 1 is written to the bit, turning the selected LED off.

Finally, in the cofunction **flashled()**, the LED to be flashed, the on time in milliseconds, and the off time in milliseconds are passed as arguments. This function uses an endless **for(;;)** loop to call the **ledon()** and **ledoff()** functions, separated by calls to the wait function **DelayMs()**. This sequence will make the indicated LED flash on and off.

As is proper in C program design, the contents of **main()** are almost trivial. The program first calls **initialize_ports()**, then begins an endless **for(;;)** loop. Within this loop, the program:

1. Calls the library function **hitwd()**, which resets the microprocessor's watchdog timer. (If the watchdog timer is not reset every so often, it will force a hard reset of the system. The purpose is to keep an intermittent program or hardware fault from locking up the system. Normally, this function is taken care of by the virtual driver, but it is called explicitly here).
2. Sets up a costatement which calls two instances of the **flashled()** function, one for each LED. Note that one LED is flashed one second on, one-half second (500 ms) off, while the other is flashed in the reverse pattern.

Note also the **wfd** keyword in the costatement. This keyword (an abbreviation for **waitfordone**, which can also be used) must be used when calling cofunctions. For a complete explanation, see Section 5 and 6 in the ***Dynamic C User's Manual***.

More Information

See the entries for the **hitwd()** and **DelayMs()** functions in the ***Dynamic C User's Manual***, as well as those for the directives **#asm** and **#endasm**. For a complete explana-

tion of how Dynamic C handles multitasking with costatements and cofunctions, see Chapter 5, "Multitasking with Dynamic C," and Chapter 6, "The Virtual Driver," in the ***Dynamic C User's Manual***.

3.1.4.3 TOGGLELED.C

One of Dynamic C's unique and powerful aspects is its ability to efficiently multitask using *cofunctions* and *costatements*. This simple application demonstrates how these program elements work.

This sample program uses two costatements to set up and manage the two tasks. Costatements must be contained in a loop that will "tap" each of them at regular intervals. This program:

1. Initializes the pins of Port A as outputs.
2. Sets all the pins of Port A high, turning off the attached LEDs.
3. Sets the toggled LED status variable **`vswitch`** to 0 (LED off).
4. Starts an endless loop using a **`while(1)`** expression, and within that loop:
   - Executes a costatement that flashes LED DS3;
   - Executes a costatement that checks the state of switch S2 and toggles the state of **`vswitch`** if it is pressed;
   - Turns LED DS2 on or off, according to the state of **`vswitch`**.

   These steps repeat as long as the program is allowed to run.

The first costatement is a compressed version of **`FLASHLED.c`**, with slightly different flash timing. It also uses the library function **`DelayMs()`** to deliver more accurate timing than the simple delay loops of the previous program.

The second costatement does more than check the status of S2. Switch contacts often "bounce" open and closed several times when the switch is actuated, and each bounce can be interpreted by fast digital logic as an independent press. To clean up this input, the code in the second costatement "debounces" the switch signal by waiting 50 milliseconds and checking the state of the switch again. If it is detected as being closed both times, the program considers it a valid switch press and toggles **`vswitch`**.

Unlike most C statements, the two costatements are not executed in their entirety on each iteration of the **`while(1)`** loop. Instead, the list of statements within each costatement is initiated on the first loop, and then executed one "slice" at a time on each successive interation. This mode of operation is known as a *state machine*, a powerful concept that permits a single processor to efficiently handle a number of independent tasks.

The ability of Dynamic C to manage state machine programs enables you to create very powerful and efficient embedded systems with much greater ease than other programming methods.

More Information

See the entries for the **`DelayMs()`** function, as well as Section 5, "Multitasking with Dynamic C," in the ***Dynamic C User's Manual***.

4. HARDWARE REFERENCE

Chapter 2 describes the hardware components and principal hardware subsystems of the RCM2200. Appendix A, "RabbitCore RCM2200 Specifications," provides complete physical and electrical specifications.

4.1 RCM2200 Digital Inputs and Outputs

Figure 4 shows the subsystems designed into the RCM2200.

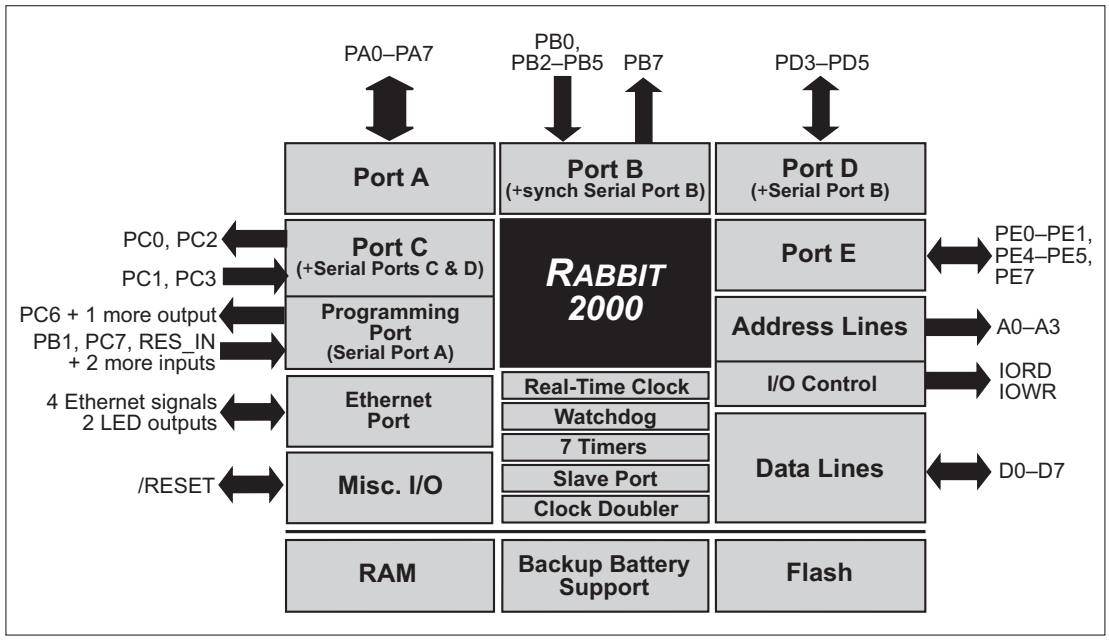


Figure 4. Rabbit Subsystems

The RCM2200 has 26 parallel I/O lines grouped in five 8-bit ports available on headers J4 and J5. The 16 bidirectional I/O lines are located on pins PA0–PA7, PD3–PD5, and PE0–PE1, PE4, PE5, and PE7. The pinouts for headers J4 and J5 are shown in Figure 5.

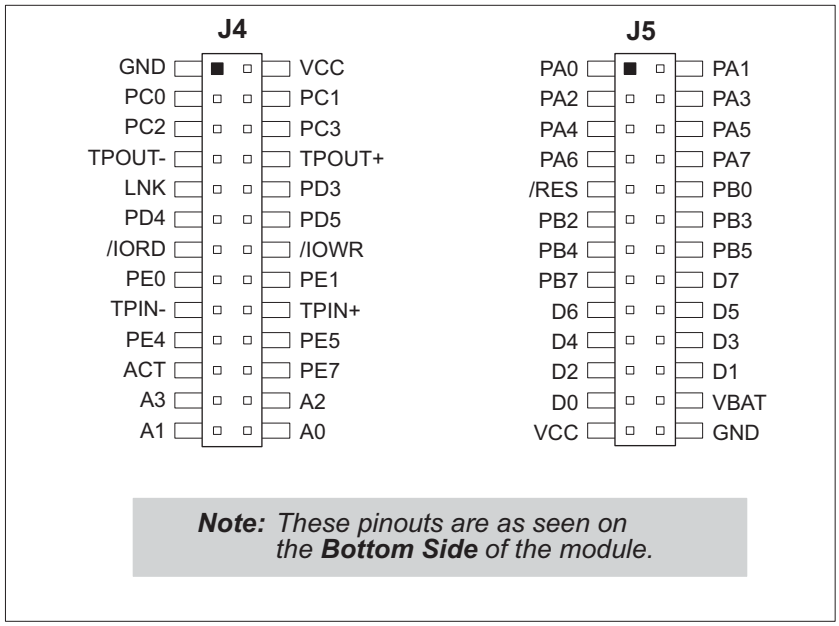


Figure 5. RCM2200 I/O Pinouts

4.1.1 Dedicated Inputs

PB0 is a general CMOS input when the Rabbit 2000 is either not using Serial Port B or is using Serial Port B in an asynchronous mode. Four other general CMOS input-only pins are located on PB2–PB5. These pins can also be used for the slave port. PB2 and PB3 are slave write and slave read strobes, while PB4 and PB5 serve as slave address lines SA0 and SA1, and are used to access the slave registers. PC1 and PC3 are general CMOS inputs only. These pins can instead be selectively enabled to serve as the serial data inputs for Serial Ports D and C.

4.1.2 Dedicated Outputs

One of the general CMOS output-only pins is located on PB7. PB7 can also be used with the slave port as the /SLAVEATTN output. This configuration signifies that the slave is requesting attention from the master. PC0 and PC2 are also output-only pins; PC0 and PC2 can instead serve as the serial data outputs for Serial Ports D and C.

4.1.3 Memory I/O Interface

Four of the Rabbit 2000 address lines (A0–A3) and all the data lines (D0–D7) are available. I/0 write (/IOWR) and I/0 read (/IORD) are also available for interfacing to external devices.

The ports on the Rabbit 2000 microprocessor used in the RCM2200 are configurable, and so the factory defaults can be reconfigured. Table 1 lists the Rabbit 2000 factory defaults and the alternate configurations.

Table 1. RCM2200 Pinout Configurations

Pin		Pin Name	Default Use	Alternate Use	Notes
Header J4	1	GND			
	2	VCC			
	3	PC0	Output	TXD	
	4	PC1	Input	RXD	
	5	PC2	Output	TXC	
	6	PC3	Input	RXC	
	7	TPOUT–			Ethernet transmit port
	8	TPOUT+			
	9	LNK			Ethernet link (LNK) LED indicator
	10	PD3	Bitwise or parallel programmable I/O		
	11	PD4		ATXB output	
	12	PD5		ARXB input	
	13	/IORD	Input (I/O read strobe)		
	14	/IOWR	Output (I/O write strobe)		
	15	PE0	Bitwise or parallel programmable I/O	I0 control or INT0A input	
	16	PE1		I1 control or INT1A input	
	17	TPIN–			Ethernet receive port
	18	TPIN+			
	19	PE4	Bitwise or parallel programmable I/O	I4 control or INT0B input	
	20	PE5		I5 control or INT1B input	
	21	ACT			Ethernet active (ACT) LED indicator
	22	PE7	Bitwise or parallel programmable I/O	I7 control or slave port chip select /SCS	
	23–26	A[3:0]			Rabbit 2000 address bus

Table 1. RCM2200 Pinout Configurations (continued)

Pin		Pin Name	Default Use	Alternate Use	Notes
Header J5	1–8	PA[0:7]	Bytewide programmable parallel I/O	Slave port data bus SD0–SD7	
	9	/RESET	Reset output	Reset input	This weak output can be driven externally
	10	PB0	Input	Serial port clock CLKB input or output	
	11	PB2	Input	Slave port write /SWR	
	12	PB3	Input	Slave port read /SRD	
	13	PB4	Input	SA0	Slave port address lines
	14	PB5	Input	SA1	
	15	PB7	Output	Slave port attention line /SLAVEATTN	
	16–23	D[7:0]	Input/Output		Rabbit 2000 data bus
	24	VBAT	3 V battery input		
	25	VCC			
	26	GND			

4.1.4 Other Inputs and Outputs

As shown in Table 1, pins PA0–PA7 can be used to allow the Rabbit 2000 to be a slave to another processor. The slave port also uses PB2–PB5, PB7, and PE7.

PE0, PE1, PE4, and PE5 can be used for up to two external interrupts. PB0 can be used to access the clock on Serial Port B of the Rabbit microprocessor. PD4 can be programmed to be a serial output for Serial Port B. PD5 can be used as a serial input by Serial Port B.

PC4, PC5, PD0, PD1, PE2, PE3, and PE6 are used for internal communication with the RealTek Ethernet interface chip.

4.2 Serial Communication

The RCM2200 board does not have an RS-232 or an RS-485 transceiver directly on the board. However, an RS-232 or RS-485 interface may be incorporated on the board the RCM2200 is mounted on. For example, the Prototyping Board supports a standard RS-232 transceiver chip.

4.2.1 Serial Ports

There are four serial ports designated as Serial Ports A, B, C, and D. All four serial ports can operate in an asynchronous mode up to the baud rate of the system clock divided by 64. An asynchronous port can handle 7 or 8 data bits. A 9th bit address scheme, where an additional bit is sent to mark the first byte of a message, is also supported. Serial Ports A and B can also be operated in the clocked serial mode. In this mode, a clock line synchronously clocks the data in or out. Either of the two communicating devices can supply the clock. When the Rabbit 2000 provides the clock, the baud rate can be up to 80% of the system clock frequency divided by 128, or 138,240 bps for a 22.1 MHz clock speed.

Serial Port A is available only on the programming port, and so is likely to be inconvenient to interface with.

4.2.2 Ethernet Port

Figure 6 shows the pinout for the RJ-45 Ethernet port (J2). Note that some Ethernet connectors are numbered in reverse to the order used here.

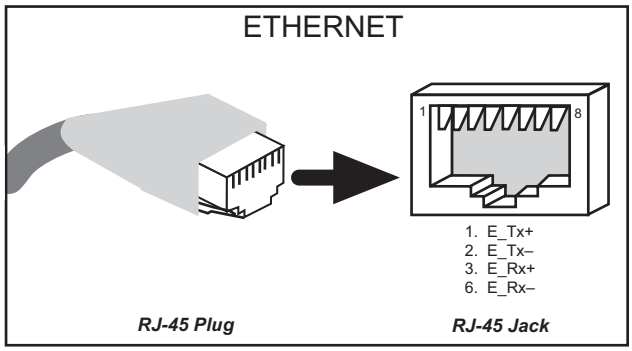


Figure 6. RJ-45 Ethernet Port Pinout

Two LEDs are placed next to the RJ-45 Ethernet jack, one to indicate an Ethernet link (**LNK**) and one to indicate Ethernet activity (**ACT**).

The Ethernet signals are also available on header J4. The **ACK** and **LNK** signals can be used to drive LEDs on the user board the RCM2200 is connected to.

The transformer/connector assembly ground is connected to the RCM2200 printed circuit board digital ground via a 0 Ω resistor, R29, as shown in Figure 7.

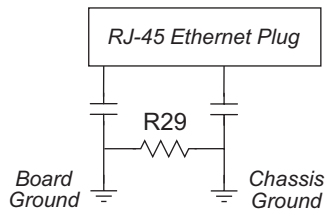


Figure 7. Isolation Resistor R29

The RJ-45 connector is shielded to minimize EMI effects to/from the Ethernet signals. Rabbit recommends that an equivalent RJ-45 connector be used on the user board if the customer wishes to have an RJ-45 connector on the user board.

> **NOTE:** The RCM2210 is available without the LEDs and the RJ-45 connector if you plan to use your own RJ-45 connector on your user board.

4.2.3 Programming Port

The RCM2200 has a 10-pin program header labeled J1. The programming port uses the Rabbit 2000's Serial Port A for communication. Dynamic C uses the programming port to download and debug programs.

The programming port is also used for the following operations.

- Cold-boot the Rabbit 2000 after a reset.
- Remotely download and debug a program over an Ethernet connection using the RabbitLink EG2110.
- Fast copy designated portions of flash memory from one Rabbit-based board (the master) to another (the slave) using the Rabbit Cloning Board.

Alternate Uses of the Serial Programming Port

All three clocked Serial Port A signals are available as

- a synchronous serial port
- an asynchronous serial port, with the clock line usable as a general CMOS input

The serial programming port may also be used as a serial port via the **DIAG** connector on the serial programming cable.

In addition to Serial Port A, the Rabbit 2000 startup-mode (SMODE0, SMODE1), status, and reset pins are available on the serial programming port.

The two startup mode pins determine what happens after a reset—the Rabbit 2000 is either cold-booted or the program begins executing at address 0x0000. These two SMODE pins can be used as general inputs once the cold boot is complete.

The status pin is used by Dynamic C to determine whether a Rabbit microprocessor is present. The status output has three different programmable functions:

1. It can be driven low on the first op code fetch cycle.
2. It can be driven low during an interrupt acknowledge cycle.
3. It can also serve as a general-purpose CMOS output.

The /RESET_IN pin is an external input that is used to reset the Rabbit 2000 and the onboard peripheral circuits on the RabbitCore module. The serial programming port can be used to force a hard reset on the RabbitCore module by asserting the /RESET_IN signal.

Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information.

4.3 Serial Programming Cable

The programming cable is used to connect the RCM2200's programming port to a PC serial COM port. The programming cable converts the RS-232 voltage levels used by the PC serial port to the TTL voltage levels used by the Rabbit 2000.

When the **PROG** connector on the programming cable is connected to the RCM2200's programming header, programs can be downloaded and debugged over the serial interface.

The **DIAG** connector of the programming cable may be used on the RCM2200's programming header with the RCM2200 operating in the Run Mode. This allows the programming port to be used as a regular serial port.

4.3.1 Changing Between Program Mode and Run Mode

The RCM2200 is automatically in Program Mode when the **PROG** connector on the programming cable is attached to the RCM2200, and is automatically in Run Mode when no programming cable is attached. When the Rabbit 2000 is reset, the operating mode is determined by the status of the SMODE pins. When the programming cable's **PROG** connector is attached, the SMODE pins are pulled high, placing the Rabbit 2000 in the Program Mode. When the programming cable's **PROG** connector is not attached, the SMODE pins are pulled low, causing the Rabbit 2000 to operate in the Run Mode.

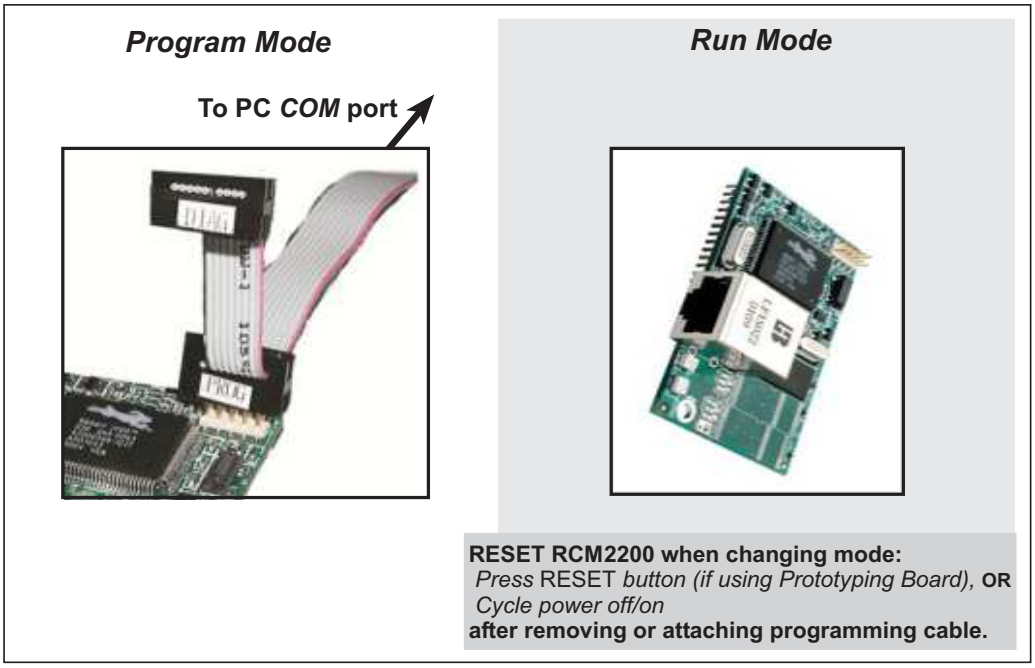


Figure 8. Switching Between Program Mode and Run Mode

A program "runs" in either mode, but can only be downloaded and debugged when the RCM2200 module is in the Program Mode.

Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information on the programming port and the programming cable.

4.3.2 Standalone Operation of the RCM2200

The RCM2200 must be programmed via the RCM2200/RCM2300 Prototyping Board or via a similar arrangement on a customer-supplied board. Once the RCM2200 has been programmed successfully, remove the programming cable from the programming connector and reset the RCM2200. The RCM2200 may be reset by cycling the power off/on or by pressing the **RESET** button on the Prototyping Board. The RCM2200 module may now be removed from the Prototyping Board for end-use installation.

CAUTION: Power to the Prototyping Board or other boards should be disconnected when removing or installing your RCM2200 module to protect against inadvertent shorts across the pins or damage to the RCM2200 if the pins are not plugged in correctly. Do not reapply power until you have verified that the RCM2200 module is plugged in correctly.

4.4 Memory

4.4.1 SRAM

The RCM2200 is designed to accept 32K to 512K of SRAM packaged in an SOIC case.

4.4.2 Flash EPROM

The RCM2200 is also designed to accept 128K to 512K of flash EPROM packaged in a TSOP case.

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

Writing to arbitrary flash memory addresses at run time is also discouraged. Instead, define a "user block" area to store persistent data. The functions **`writeUserBlock`** and **`readUserBlock`** are provided for this.

A Flash Memory Bank Select jumper configuration option based on 0 Ω surface-mounted resistors exists at JP2, JP3, and JP5 (corresponding to the flash memory chips at U8 [second flash on RCM2250], U3 [RCM2200], and U7 [no flash installed on existing RCM2200 versions]). 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.

> **NOTE:** Only the Normal Mode, which corresponds to using the full code space, is supported at the present time.

4.4.3 Dynamic C BIOS Source Files

The Dynamic C BIOS source files handle different standard RAM and flash EPROM sizes automatically.

4.5 Other Hardware

4.5.1 Clock Doubler

The RCM2200 takes advantage of the Rabbit 2000 microprocessor's internal clock doubler. A built-in clock doubler allows half-frequency crystals to be used to reduce radiated emissions. The 22.1 MHz frequency is generated using an 11.0592 MHz crystal. The clock doubler is disabled automatically in the BIOS for crystals with a frequency above 12.9 MHz.

The clock doubler may be disabled if 22.1 MHz clock speeds are not required. Disabling the Rabbit 2000 microprocessor's internal clock doubler will reduce power consumption and further reduce radiated emissions. The clock doubler is disabled with a simple configuration macro as shown below.

1. Select the "Defines" tab from the Dynamic C **Options > Project Options** menu.
2. Add the line **CLOCK_DOUBLED=0** to always disable the clock doubler.

   The clock doubler is enabled by default, and usually no entry is needed. If you need to specify that the clock doubler is always enabled, add the line **CLOCK_DOUBLED=1** to always enable the clock doubler. The clock speed will be doubled as long as the crystal frequency is less than or equal to 26.7264 MHz.
3. Click **OK** to save the macro. The clock doubler will now remain off whenever you are in the project file where you defined the macro.

4.5.2 Spectrum Spreader

RCM2200 RabbitCore modules that have a Rabbit 2000 microprocessor labeled ***IQ4T*** (or higher) are equipped with a Rabbit 2000 microprocessor that has a spectrum spreader, which helps to mitigate EMI problems. By default, the spectrum spreader is on automatically for RCM2200 modules that carry the ***IQ4T*** (or higher) marking when used with Dynamic C 7.30 or later versions, but the spectrum spreader may also be turned off or set to a stronger setting. The means for doing so is through a simple configuration macro as shown below.

1. Select the "Defines" tab from the Dynamic C **Options > Project Options** menu.
2. Normal spreading is the default, and usually no entry is needed. If you need to specify normal spreading, add the line

   ```
   ENABLE_SPREADER=1
   ```

   For strong spreading, add the line

   ```
   ENABLE_SPREADER=2
   ```

   To disable the spectrum spreader, add the line

   ```
   ENABLE_SPREADER=0
   ```

   NOTE: The strong spectrum-spreading setting is usually not necessary for the RCM2200.
3. Click **OK** to save the macro. The spectrum spreader will now remain off whenever you are in the project file where you defined the macro.

There is no spectrum spreader functionality for RCM2200 RabbitCore modules that have a Rabbit 2000 microprocessor labeled ***IQ1T***, ***IQ2T***, or ***IQ3T***, or when using any RCM2200 with a version of Dynamic C prior to 7.30.

5. SOFTWARE REFERENCE

Dynamic C is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Rabbit single-board computers and other single-board computers based on the Rabbit microprocessor. Chapter 4 provides the libraries and function calls related to the RCM2200.

5.1 More About Dynamic C

Dynamic C has been in use worldwide since 1989. Dynamic C is specially designed for programming embedded systems, and features quick compile and interactive debugging. A complete reference to Dynamic C is contained in the ***Dynamic C User's Manual***.

You have a choice of doing your software development in the flash memory or in the static RAM included on the RCM2200. The flash memory and SRAM options are selected with the **Options > Project Options > Compiler** menu.

The advantage of working in RAM is to save wear on the flash memory, which is limited to about 100,000 write cycles. The disadvantage is that the code and data might not both fit in RAM.

> **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.
>
> **NOTE:** Do not depend on the flash memory sector size or type. Due to the volatility of the flash memory market, the RCM2200 and Dynamic C were designed to accommodate flash devices with various sector sizes.

RCM2250 and RCM2260 RabbitCore modules have two 256K flash memories. By default, Dynamic C will use only the first flash memory for program code in these RCM2250 and RCM2260 RabbitCore modules. Uncomment he BIOS `USE_2NDFLASH_CODE` macro to allow the second flash memory to hold any program code that is in excess of the available memory in the first flash.

Developing software with Dynamic C is simple. Users can write, compile, and test C and assembly code without leaving the Dynamic C development environment. Debugging occurs while the application runs on the target. Alternatively, users can compile a program to an image file for later loading. Dynamic C runs on PCs under Windows 95 or later. Programs can be downloaded at baud rates of up to 460,800 bps after the program compiles.

Dynamic C has a number of standard features:

- Full-feature source and/or assembly-level debugger, no in-circuit emulator required.
- Royalty-free TCP/IP stack with source code and most common protocols.
- Hundreds of functions in source-code libraries and sample programs:
  - Exceptionally fast support for floating-point arithmetic and transcendental functions.
  - RS-232 and RS-485 serial communication.
  - Analog and digital I/O drivers.
  - I^2C, SPI, GPS, file system.
  - LCD display and keypad drivers.
- Powerful language extensions for cooperative or preemptive multitasking.
- Loader utility program to load binary images into Rabbit targets in the absence of Dynamic C.
- Provision for customers to create their own source code libraries and augment on-line help by creating "function description" block comments using a special format for library functions.
- Standard debugging features:
  - Breakpoints—Set breakpoints that can disable interrupts.
  - Single-stepping—Step into or over functions at a source or machine code level, µC/OS-II aware.
  - Code disassembly—The disassembly window displays addresses, opcodes, mnemonics, and machine cycle times. Switch between debugging at machine-code level and source-code level by simply opening or closing the disassembly window.
  - Watch expressions—Watch expressions are compiled when defined, so complex expressions including function calls may be placed into watch expressions. Watch expressions can be updated with or without stopping program execution.
  - Register window—All processor registers and flags are displayed. The contents of general registers may be modified in the window by the user.
  - Stack window—shows the contents of the top of the stack.
  - Hex memory dump—displays the contents of memory at any address.
  - **STDIO** window—`printf` outputs to this window and keyboard input on the host PC can be detected for debugging purposes. `printf` output may also be sent to a serial port or file.

5.2 I/O

The RCM2200 was designed to interface with other systems, and so there are no drivers written specifically for the I/O. The general Dynamic C read and write functions allow you to customize the parallel I/O to meet your specific needs. For example, use

```
WrPortI(PEDDR, &PEDDRShadow, 0x00);
```

to set all the port E bits as inputs, or use

```
WrPortI(PEDDR, &PEDDRShadow, 0xFF);
```

to set all the Port E bits as outputs.

The sample programs in the Dynamic C **SAMPLES\RCM2200** directory provide further examples.

5.2.1 PCLK Output

The PCLK output is controlled by bits 7 and 6 of the Global Output Register (GOCR) on the Rabbit 2000 microprocessor, and so can be enabled or disabled in software. Starting with Dynamic C v 7.02, the PCLK output is disabled by default at compile time to minimize radiated emissions; the PCLK output is enabled in earlier versions of Dynamic C.

Use the following code to set the PCLK output as needed.

PCLK output driven with peripheral clock:

```
WrPortI(GOCR, &GOCRShadow, (GOCRShadow&~0xc0));
```

PCLK output driven with peripheral clock ÷ 2:

```
WrPortI(GOCR, &GOCRShadow, ((GOCRShadow&~0xc0)| 0x40));
```

PCLK output off (low):

```
WrPortI(GOCR, &GOCRShadow, ((GOCRShadow&~0xc0)| 0x80));
```

PCLK output on (high):

```
WrPortI(GOCR, &GOCRShadow, (GOCRShadow | 0xc0));
```

5.2.2 External Interrupts

The Rabbit 2000 microprocessor has four external interrupt inputs on Parallel Port E, which is accessed through pins PE0, PE1, PE4, and PE5 on header J4. These pins may be used either as I/O ports or as external interrupt inputs.

Earlier versions of the Rabbit 2000 microprocessor labeled ***IQ1T*** or ***IQ2T*** would occasionally lose an interrupt request when one of the interrupt inputs was used as a pulse counter. See Technical Note 301, ***Rabbit 2000 Microprocessor Interrupt Problem***, for further information on how to work around this problem if you purchased your RCM2200 before July, 2002, and the Rabbit 2000 microprocessor is labeled ***IQ1T*** or ***IQ2T***.

NOTE: Interrupts on RCM2000 series RabbitCore modules sold after July, 2002, work correctly and do not need this workaround.

5.3 Serial Communication Drivers

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 User's Manual*** and Technical Note 213, ***Rabbit 2000 Serial Port Software***.

5.4 TCP/IP Drivers

The TCP/IP drivers are located in the **TCPIP** directory. Complete information on these libraries and the TCP/IP functions is provided in the ***Dynamic C TCP/IP User's Manual***.

5.5 Upgrading Dynamic C

Dynamic C patches that focus on bug fixes are available from time to time. Check the Web site www.rabbit.com/support/ for the latest patches, workarounds, and bug fixes.

The default installation of a patch or bug fix is to install the file in a directory (folder) different from that of the original Dynamic C installation. Rabbit recommends using a different directory so that you can verify the operation of the patch without overwriting the existing Dynamic C installation. If you have made any changes to the BIOS or to libraries, or if you have programs in the old directory (folder), make these same changes to the BIOS or libraries in the new directory containing the patch. Do ***not*** simply copy over an entire file since you may overwrite a bug fix; of course, you may copy over any programs you have written. Once you are sure the new patch works entirely to your satisfaction, you may retire the existing installation, but keep it available to handle legacy applications.

5.5.1 Extras

Dynamic C installations are designed for use with the board they are included with, and are included at no charge as part of our low-cost kits.

Starting with Dynamic C version 9.60, Dynamic C includes the popular µC/OS-II real-time operating system, point-to-point protocol (PPP), FAT file system, RabbitWeb, and other select libraries. Rabbit also offers for purchase the Rabbit Embedded Security Pack featuring the Secure Sockets Layer (SSL) and a specific Advanced Encryption Standard (AES) library.

In addition to the Web-based technical support included at no extra charge, a one-year telephone-based technical support subscription is also available for purchase.

Visit our Web site at www.rabbit.com for further information and complete documentation.

6. USING THE TCP/IP FEATURES

6.1 TCP/IP Connections

Programming and development can be done with the RCM2200 RabbitCore modules without connecting the Ethernet port to a network. However, if you will be running the sample programs that use the Ethernet capability or will be doing Ethernet-enabled development, you should connect the RCM2200 module's Ethernet port at this time.

Before proceeding you will need to have the following items.

- If you don't have Ethernet access, you will need at least a 10Base-T Ethernet card (available from your favorite computer supplier) installed in a 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 Rabbit in a TCP/IP tool kit. More information is available at www.rabbit.com.

1. Connect the AC adapter and the programming cable as shown in Chapter 2, "Getting Started."
2. Ethernet Connections

   There are four options for connecting the RCM2200 module to a network for development and runtime purposes. The first two options permit total freedom in selecting network addresses and use of the "network," as no action can interfere with other users. We recommend one of these options for initial development.

- **No LAN —** The simplest alternative for desktop development. Connect the RCM2200's Ethernet port directly to the PC's network interface card using an RJ-45 *crossover cable*. A crossover cable is a special cable that flips some connections between the two connectors and permits direct connection of two client systems. A standard RJ-45 network cable will not work for this purpose.
- **Micro-LAN —** Another simple alternative for desktop development. Use a small Ethernet 10Base-T hub and connect both the PC's network interface card and the RCM2200's Ethernet port to it, using standard network cables.

The following options require more care in address selection and testing actions, as conflicts with other users, servers and systems can occur:

- **LAN —** Connect the RCM2200's Ethernet port to an existing LAN, preferably one to which the development PC is already connected. You will need to obtain IP addressing information from your network administrator.
- **WAN —** The RCM2200 is capable of direct connection to the Internet and other Wide Area Networks, but exceptional care should be used with IP address settings and all network-related programming and development. We recommend that development and debugging be done on a local network before connecting a RabbitCore system to the Internet.

  TIP: Checking and debugging the initial setup on a micro-LAN is recommended before connecting the system to a LAN or WAN.

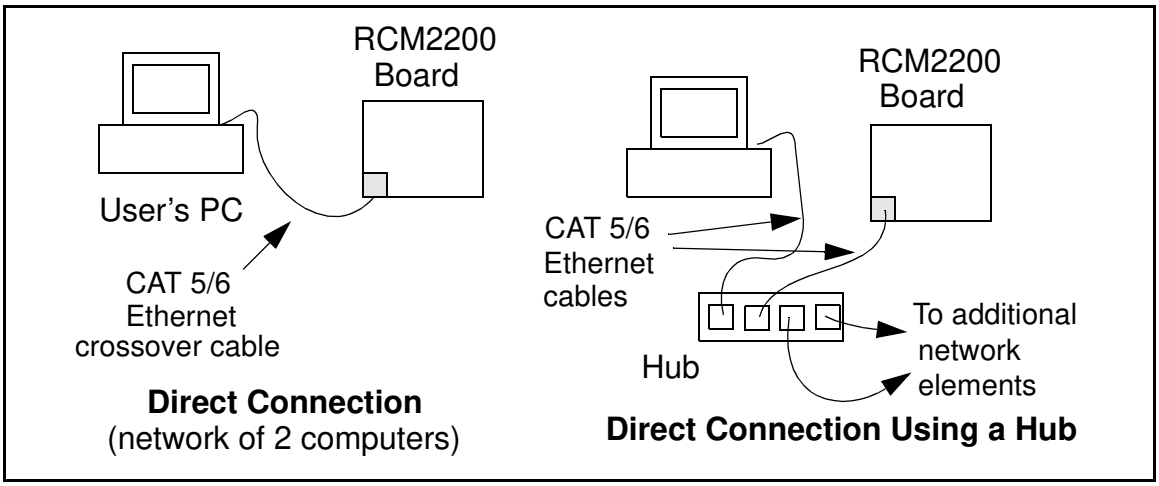
The PC running Dynamic C through the serial port on the RCM2200 does not need to be the PC with the Ethernet card.

3. Apply Power

   Plug in the AC adapter. The RCM2200 is now ready to be used.

6.2 Running 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 the user connect his PC and the RCM2200 board 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.



Obtaining IP addresses to interact over an existing, operating, network can involve a number of complications, and must usually be done with cooperation from your ISP and/or network systems administrator (if your company has one). For this reason, it is suggested that the user begin instead by using a direct connection between a PC and the RCM2200 board using an Ethernet crossover cable or a simple arrangement with a hub. (A crossover cable should not be confused with regular straight through cables.) The hub and a wide variety of cables can also be purchased from a local computer store.

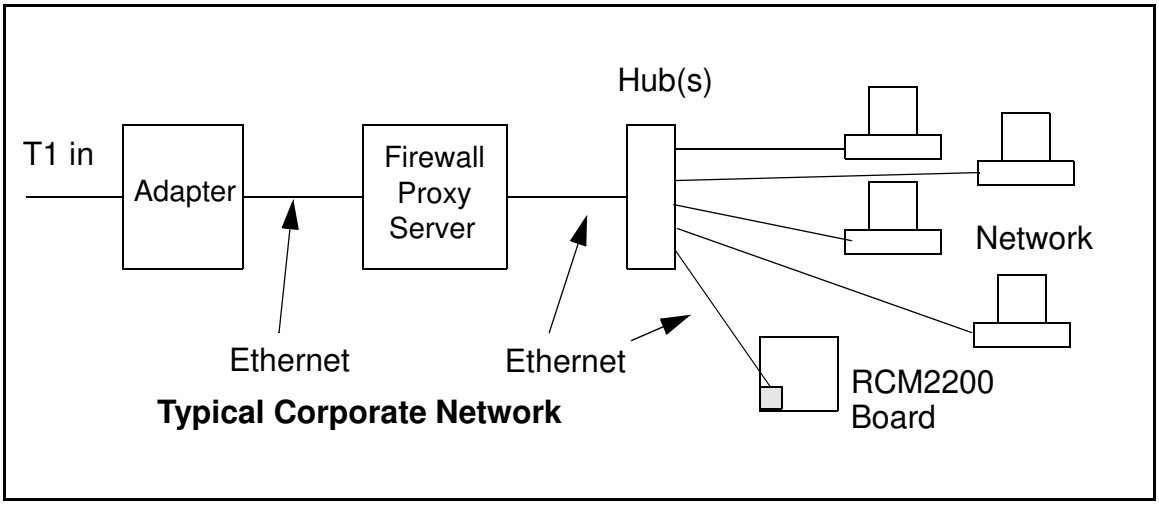
In order to set up this direct connection, the user will have to use a PC without networking, or disconnect a PC from the corporate network, or install a second Ethernet adapter and set up a separate private network attached to the second Ethernet adapter. Disconnecting your PC from the corporate network may be easy or nearly impossible, depending on how it is set up. Mobile PCs, such as laptops, are designed to be connected and disconnected, and will present the least problem. If your PC boots from the network or is dependent on the network for some or all of its disks, then it probably should not be disconnected. If a second Ethernet adapter is used, be aware that Windows TCP/IP will send messages to one adapter or the other, depending on the IP address and the binding order in Microsoft products. Thus you should have different ranges of IP addresses on your private network from those used on the corporate network. If both networks service the same IP address, then Windows may send a packet intended for your private network to the corporate network. A similar situation will take place if you use a dial-up line to send a packet to the Internet. Windows may try to send it via the local Ethernet network if it is also valid for that network.

The following IP addresses are set aside for local networks and are not allowed on the Internet: 10.0.0.0 to 10.255.255.255, 172.16.0.0 to 172.31.255.255, and 192.168.0.0 to 192.168.255.255.

The RCM2200 board uses a 10Base-T type of Ethernet connection, which is the most common scheme. The RJ-45 connectors are similar to U.S. style telephone connectors, except they are larger and have 8 contacts.

An alternative to the direct connection using a crossover cable is a direct connection using a hub. The hub relays packets received on any port to all of the ports on the hub. Hubs are low in cost and are readily available. The RCM2200 board uses 10 Mbps Ethernet, so the hub or Ethernet adapter must be either a 10 Mbps unit or a 10/100 unit that adapts to either 10 or 100 Mbps.

In a corporate setting where the Internet is brought in via a high-speed line, there are typically machines between the outside Internet and the internal network. These machines include a combination of proxy servers and firewalls that filter and multiplex Internet traffic. In the configuration below, the RCM2200 board could be given a fixed address so any of the computers on the local network would be able to contact it. It may be possible to configure the firewall or proxy server to allow hosts on the Internet to directly contact the controller, but it would probably be easier to place the controller directly on the external network outside of the firewall. This avoids some of the configuration complications by sacrificing some security.



Typical Corporate Network

If your system administrator can give you an Ethernet cable along with its IP address, the netmask and the gateway address, then you may be able to run the sample programs without having to setup a direct connection between your computer and the RCM2200 board. You will also need the IP address of the nameserver, the name or IP address of your mail server, and your domain name for some of the sample programs.

6.3 IP Addresses Explained

IP (Internet Protocol) addresses are expressed as 4 decimal numbers separated by periods, for example:

216.103.126.155

10.1.1.6

Each decimal number must be between 0 and 255. The total IP address is a 32-bit number consisting of the 4 bytes expressed as shown above. A local network uses a group of adjacent IP addresses. There are always 2^N IP addresses in a local network. The netmask (also called subnet mask) determines how many IP addresses belong to the local network. The netmask is also a 32-bit address expressed in the same form as the IP address. An example netmask is:

255.255.255.0

This netmask has 8 zero bits in the least significant portion, and this means that 2^8 addresses are a part of the local network. Applied to the IP address above (216.103.126.155), this netmask would indicate that the following IP addresses belong to the local network:

216.103.126.0

216.103.126.1

216.103.126.2

etc.

216.103.126.254

216.103.126.255

The lowest and highest address are reserved for special purposes. The lowest address (216.103.126.0) is used to identify the local network. The highest address (216.103.126.255) is used as a broadcast address. Usually one other address is used for the address of the gateway out of the network. This leaves 256 - 3 = 253 available IP addresses for the example given.

6.4 How IP Addresses are Used

The actual hardware connection via an Ethernet uses Ethernet adapter addresses (also called MAC addresses.) These are 48-bit addresses and are unique for every Ethernet adapter manufactured. In order to send a packet to another computer, given the IP address of the other computer, it is first determined if the packet needs to be sent directly to the other computer or to the gateway. In either case, there is an IP address on the local network to which the packet must be sent. A table is maintained to allow the protocol driver to determine the MAC address corresponding to a particular IP address. If the table is empty, the MAC address is determined by sending an Ethernet broadcast packet to all devices on the local network asking the device with the desired IP address to answer with its MAC address. In this way, the table entry can be filled in. If no device answers, then the device is nonexistent or inoperative, and the packet cannot be sent.

IP addresses are arbitrary and can be allocated as desired provided that they don't conflict with other IP addresses. However, if they are to be used with the Internet, then they must be numbers that are assigned to your connection by proper authorities, generally by delegation via your service provider.

6.5 Dynamically Assigned Internet Addresses

In many instances, there are no fixed IP addresses. This is the case when, for example, you are assigned an IP address dynamically by your dial-up Internet service provider (ISP) or when you have a device that provides your IP addresses using the Dynamic Host Configuration Protocol (DHCP). The RCM2200 can use such IP addresses to send and receive packets on the Internet, but you must take into account that this IP address may only be valid for the duration of the call or for a period of time, and could be a private IP address that is not directly accessible to others on the Internet. These private address can be used to perform some Internet tasks such as sending e-mail or browsing the Web, but usually cannot be used to participate in conversations that originate elsewhere on the Internet. If you want to find out this dynamically assigned IP address, under Windows XP you can run the **ipconfig** program while you are connected and look at the interface used to connect to the Internet.

Many networks use private IP addresses that are assigned using DHCP. When your computer comes up, and periodically after that, it requests its networking information from a DHCP server. The DHCP server may try to give you the same address each time, but a fixed IP address is usually not guaranteed.

If you are not concerned about accessing the RCM2200 from the Internet, you can place the RCM2200 on the internal network using a private address assigned either statically or through DHCP.

6.6 Placing Your Device on the Internet

In many corporate settings, users are isolated from the Internet by a firewall and/or a proxy server. These devices attempt to secure the company from unauthorized network traffic, and usually work by disallowing traffic that did not originate from inside the network. If you want users on the Internet to communicate with your RCM2200, you have several options. You can either place the RCM2200 directly on the Internet with a real Internet address or place it behind the firewall. If you place the RCM2200 behind the firewall, you need to configure the firewall to translate and forward packets from the Internet to the RCM2200.

6.7 How to Set IP Addresses in the 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 Coyote 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.

With the introduction of Dynamic C 7.30 we have taken steps to make it easier to run many of our sample programs. You will see a **TCPCONFIG** macro. This macro tells Dynamic C to select your configuration from a list of default configurations. You will have three choices when you encounter a sample program with the **TCPCONFIG** macro.

1. You can replace the **TCPCONFIG** macro with individual **MY_IP_ADDRESS**, **MY_NETMASK**, **MY_GATEWAY**, and **MY_NAMESERVER** macros in each program.
2. You can leave **TCPCONFIG** at the usual default of 1, which will set the IP configurations to **10.10.6.100**, the netmask to **255.255.255.0**, and the nameserver and gateway to **10.10.6.1**. If you would like to change the default values, for example, to use an IP address of **10.1.1.2** for the Coyote board, and **10.1.1.1** for your PC, you can edit the values in the section that directly follows the "General Configuration" comment in the **TCP_CONFIG.LIB** library. You will find this library in the **LIB\TCPIP** directory.
3. You can create a **CUSTOM_CONFIG.LIB** library and use a **TCPCONFIG** value greater than 100. Instructions for doing this are at the beginning of the **TCP_CONFIG.LIB** library in the **LIB\TCPIP** directory.

There are some other "standard" configurations for **TCPCONFIG** that let you select different features such as DHCP. Their values are documented at the top of the **TCP_CONFIG.LIB** library in the **LIB\TCPIP** directory. More information is available in the ***Dynamic C TCP/IP User's Manual***.

IP Addresses Before Dynamic C 7.30

Most of the sample programs use macros to define the IP address assigned to the board and the IP address of the gateway, if there is a gateway. Instead of the **TCPCONFIG** macro, you will see a **MY_IP_ADDRESS** macro and other macros.

```
#define MY_IP_ADDRESS "10.10.6.170"
#define MY_NETMASK "255.255.255.0"
#define MY_GATEWAY "10.10.6.1"
#define MY_NAMESERVER "10.10.6.1"
```

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

```
#define MY_IP_ADDRESS "10.1.1.2"
#define MY_NETMASK "255.255.255.0"
// #define MY_GATEWAY "10.10.6.1"
// #define MY_NAMESERVER "10.10.6.1"
```

In this case, the gateway and nameserver are not used, and are 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**.

6.8 How to Set Up Your Computer for Direct Connect

Follow these instructions to set up your PC or notebook. Check with your administrator if you are unable to change the settings as described here since you may need administrator privileges. The instructions are specifically for Windows 2000, but the interface is similar for other versions of Windows.

> **TIP:** If you are using a PC that is already on a network, you will disconnect the PC from that network to run these sample programs. Write down the existing settings before changing them to facilitate restoring them when you are finished with the sample programs and reconnect your PC to the network.

1. Go to the control panel (**Start > Settings > Control Panel**), and then double-click the Network icon.
2. Select the network interface card used for the Ethernet interface you intend to use (e.g., **TCP/IP Xircom Credit Card Network Adapter**) and click on the "Properties" button. Depending on which version of Windows your PC is running, you may have to select the "Local Area Connection" first, and then click on the "Properties" button to bring up the Ethernet interface dialog. Then "Configure" your interface card for a "10Base-T Half-Duplex" or an "Auto-Negotiation" connection on the "Advanced" tab.

   > **NOTE:** Your network interface card will likely have a different name.

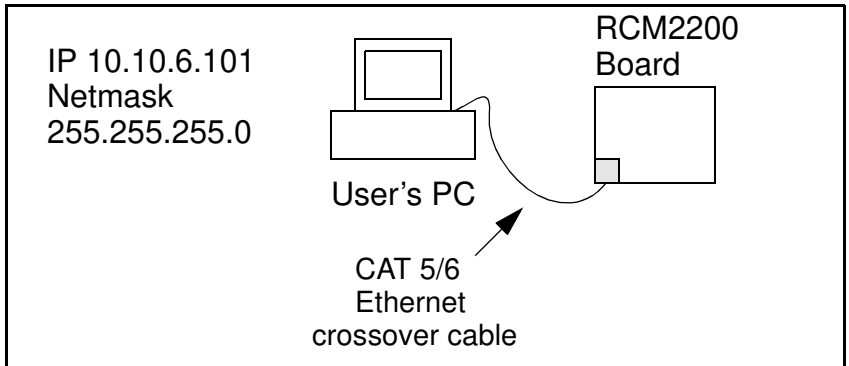
3. Now select the **IP Address** tab, and check **Specify an IP Address**, or select TCP/IP and click on "Properties" to assign an IP address to your computer (this will disable "obtain an IP address automatically"):

   IP Address : 10.10.6.101

   Netmask : 255.255.255.0

   Default gateway : 10.10.6.1

4. Click **<OK>** or **<Close>** to exit the various dialog boxes.



Direct Connection PC to RCM2200 Board

6.9 Run the PINGME.C Sample Programs

Connect the crossover cable from your computer's Ethernet port to the RCM2200 board's RJ-45 Ethernet connector. Open this sample program from the **SAMPLES\TCPIP\ICMP** folder, compile the program, and start it running under Dynamic C. When the program starts running, the green **LNK** light on the RCM2200 board should be on to indicate 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 pingme program:

```
ping 10.10.6.100
```

or by **Start > Run**

and typing the entry

```
ping 10.10.6.100
```

Notice that the red **ACT** light flashes on the RCM2200 board 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.

6.10 Running More Sample Programs With Direct Connect

The sample programs discussed here are in the Dynamic C **SAMPLES\RCM2200** folder.

- **CONSOLE.C**—Demonstrates the features of **ZCONSOLE.LIB** command-oriented console library. This program is also run in conjunction with **SERDCLIENT.C** or **SPCLIENT.C**.
- **ETHCORE1.C**—Creates two "devices" (lights) and two "buttons" in the Web browser to toggle them. Users can change the status of the lights. If the RCM2200 is plugged into the **MASTER** slot on the Prototyping Board, the lights on the Prototyping Board will track the ones in the Web browser. As long as you have not modified the **TCPCONFIG 1** macro in the sample program, enter the following server address in your Web browser to bring up the Web page served by the sample program.

  http://10.10.6.100

  Otherwise use the TCP/IP settings you entered in the **TCP_CONFIG.LIB** library.
- **MYECHO.C**—Operates RCM2200 as a basic server. When a client connects, echoes back any data sent by the client.
- **SERDCLIENT.C**—Demonstrates the ability of a Rabbit-based target board to update files on the Web server of the RCM2200 board it is connected to via Serial Port D. This program is run in conjunction with **CONSOLE.C**.
- **SPCLIENT.C**—Demonstrates the ability of a Rabbit-based target board to update files on the Web server of the RCM2200 board it is connected to via the slave port. This program is run in conjunction with **CONSOLE.C**.

6.11 Where Do I Go From Here?

NOTE: If you purchased your RCM2200 through a distributor or through a Rabbit partner, contact the distributor or partner first for technical support.

If there are any problems at this point:

- Use the Dynamic C **Help** menu to get further assistance with Dynamic C.
- Check the Rabbit Technical Bulletin Board and forums at www.rabbit.com/support/bb/ and at www.rabbit.com/forums/.
- Use the Technical Support e-mail form at www.rabbit.com/support/.

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***.

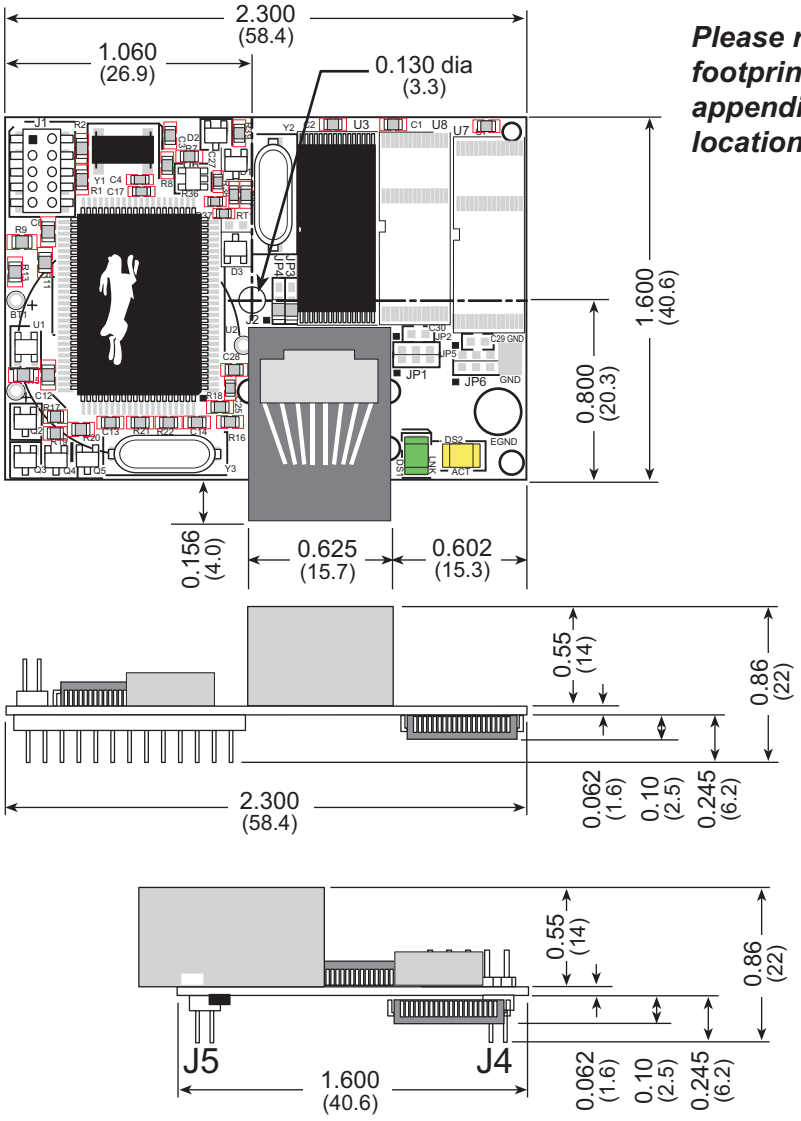
Please refer to the ***Dynamic C TCP/IP User's Manual*** to develop your own applications. ***An Introduction to TCP/IP*** provides background information on TCP/IP, and is available on the CD and on our Web site.

APPENDIX A. RABBITCORE RCM2200 SPECIFICATIONS

Appendix A provides the specifications for the RCM2200, and describes the conformal coating.

A.1 Electrical and Mechanical Characteristics

Figure A-1 shows the mechanical dimensions for the RCM2200.



Please refer to the RCM2200 footprint diagram later in this appendix for precise header locations.

Figure A-1. RCM2200 Dimensions

NOTE: All measurements are in inches followed by millimeters enclosed in parentheses. All dimensions have a manufacturing tolerance of ±0.01" (0.25 mm).

It is recommended that you allow for an “exclusion zone” of 0.04" (1 mm) around the RCM2200 in all directions when the RCM2200 is incorporated into an assembly that includes other printed circuit boards. An “exclusion zone” of 0.16" (4 mm) is recommended below the RCM2200 when the RCM2200 is plugged into another assembly using the shortest connectors for headers J4 and J5. Figure A-2 shows this “exclusion zone.”

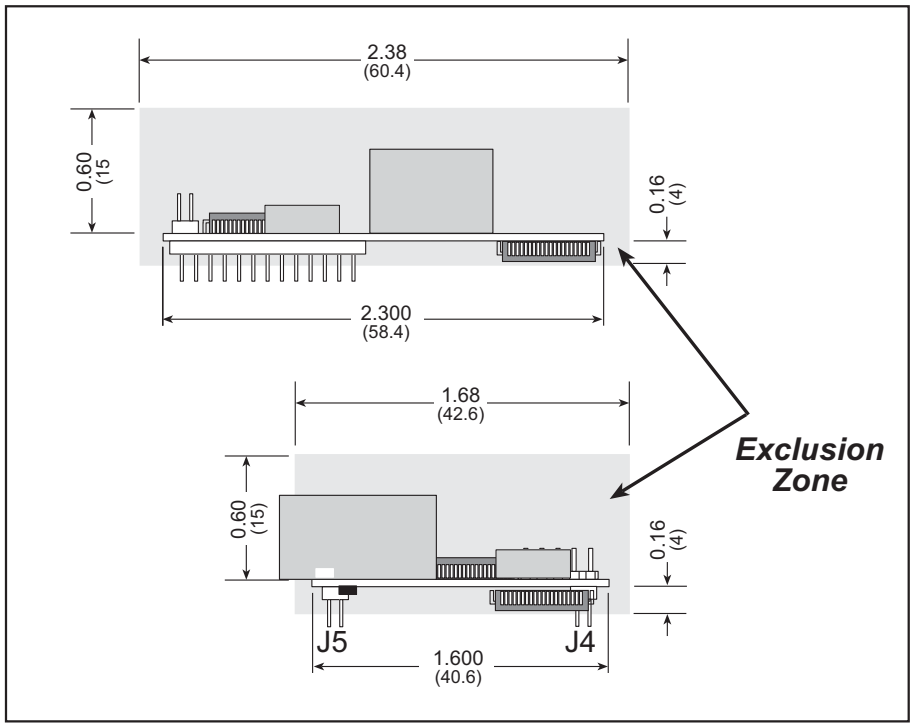


Figure A-2. RCM2200 “Exclusion Zone”

Table A-1 lists the electrical, mechanical, and environmental specifications for the RCM2200.

Table A-1. RabbitCore RCM2200 Specifications

Parameter	RCM2200	RCM2210	RCM2250	RCM2260
Microprocessor	Rabbit 2000® at 22.1 MHz			
Ethernet Port (10/100-compatible with 10Base-T interface)	RJ-45, 2 LEDs	Raw signals only	RJ-45, 2 LEDs	Raw signals only
Flash Memory	One 256K		Two 256K	Two 256K
SRAM	128K		512K	512K
Backup Battery	Connection for user-supplied backup battery (to support RTC and SRAM)			
General-Purpose I/O	26 parallel I/0 lines grouped in five 8-bit ports (shared with serial ports): • 16 configurable I/O • 7 fixed inputs • 3 fixed outputs			
Additional Inputs	2 startup mode, reset			
Additional Outputs	Status, reset			
Memory, I/O Interface	4 address lines, 8 data lines, I/O read/write			
Serial Ports	Four 5 V CMOS-compatible ports. Two ports are configurable as clocked ports, one is a dedicated RS-232 programming port.			
Serial Rate	Maximum burst rate = CLK/32 Maximum sustained rate = CLK/64			
Slave Interface	A slave port allows the RCM2200 to be used as an intelligent peripheral device slaved to a master processor, which may either be another Rabbit 2000 or any other type of processor			
Real-Time Clock	Yes			
Timers	Five 8-bit timers cascadable in pairs, one 10-bit timer with 2 match registers that each have an interrupt			
Watchdog/Supervisor	Yes			
Power	4.75 V to 5.25 V DC, 134 mA			
Operating Temperature	–40°C to +70°C			
Humidity	5% to 95%, noncondensing			
Connectors	Two IDC headers 2 × 13, 2 mm pitch			
Board Size	1.60" × 2.30" × 0.86" (41 mm × 59 mm × 22 mm)			

A.1.1 Headers

The RCM2200 uses headers at J4 and J5 for physical connection to other boards. J4 and J5 are 2 × 13 SMT headers with a 2 mm pin spacing. J1, the programming port, is a 2 × 5 header with a 2 mm pin spacing.

Figure A-3 shows the layout of another board for the RCM2200 to be plugged into. These values are relative to the header connectors.

A.1.2 Physical Mounting

A 9/32" (7 mm) standoff with a 4-40 screw is recommended to attach the RCM2200 to a user board at the hole position shown in Figure A-3. Either use plastic hardware, or use insulating washers to keep any metal hardware from shorting out signals on the RCM2200.

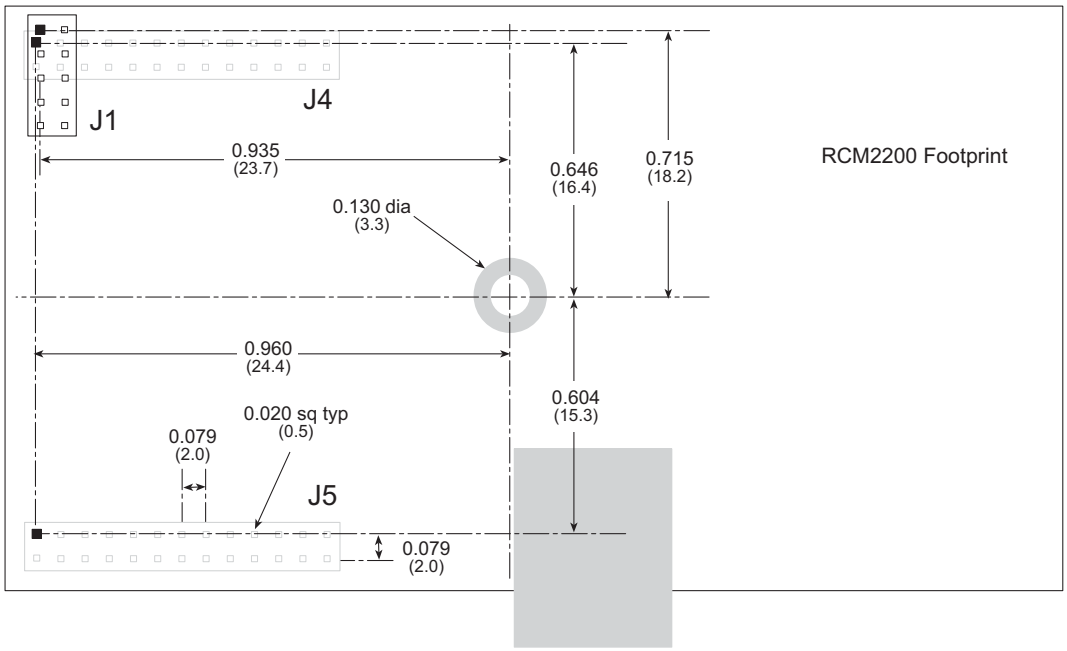


Figure A-3. User Board Footprint for RCM2200

A.2 Bus Loading

You must pay careful attention to bus loading when designing an interface to the RCM2200. This section provides bus loading information for external devices.

Table A-2 lists the capacitance for the various RCM2200 I/O ports.

Table A-2. Capacitance of Rabbit 2000 I/O Ports

I/O Ports	Input Capacitance (pF)	Output Capacitance (pF)
Parallel Ports A to E	12	14
Data Lines BD0–BD7	10	12
Address Lines BA0–BA12	4	8

Figure A-4 shows a typical timing diagram for the Rabbit 2000 microprocessor external I/O read and write cycles.

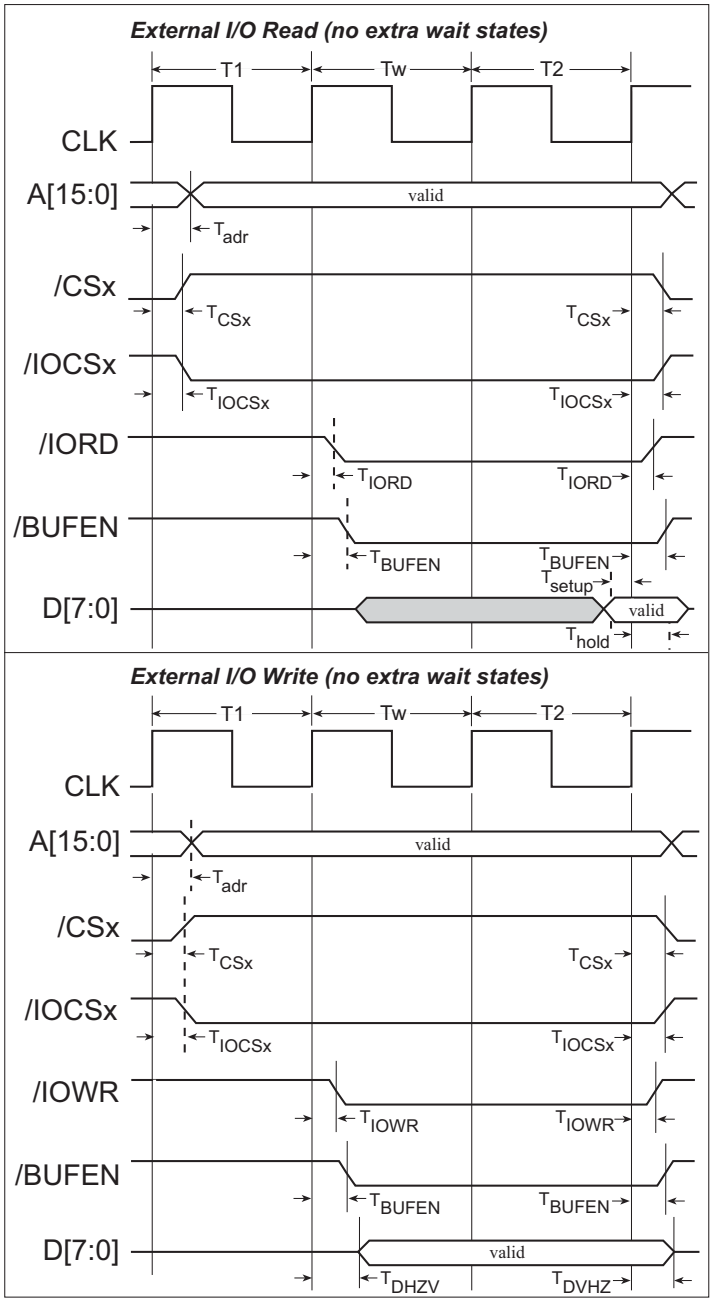


Figure A-4. External I/O Read and Write Cycles—No Extra Wait States

T_{adr} is the time required for the address output to reach 0.8 V. This time depends on the bus loading. T_{setup} is the data setup time relative to the clock. Tsetup is specified from 30%/70% of the V_{DD} voltage level.

Table A-3 lists the parameters shown in these figures and provides minimum or measured values.

Table A-3. Memory and External I/O Read/Write Parameters

Parameter		Description	Value	
Read Parameters	T_{adr}	Time from CPU clock rising edge to address valid	Max.	7 ns @ 20 pF, 5 V 14 ns @ 70 pF, 5 V
	T_{setup}	Data read setup time	Min.	2 ns @ 5 V
	T_{hold}	Data read hold time	Min.	0 ns
Write Parameters	T_{adr}	Time from CPU clock rising edge to address valid	Max.	7 ns @ 20 pF, 5 V 14 ns @ 70 pF, 5 V
	T_{hold}	Data write hold time from /WEx or /IOWR	Min.	½ CPU clock cycle

A.3 Rabbit 2000 DC Characteristics

Table A-4 outlines the DC characteristics for the Rabbit 2000 at 5.0 V over the recommended operating temperature range from T_a = –40°C to +85°C, V_{DD} = 4.5 V to 5.5 V.

Table A-4. 5.0 Volt DC Characteristics

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
I_{IH}	Input Leakage High	$V_{IN} = V_{DD}$, V_{DD} = 5.5 V			10	µA
I_{IL}	Input Leakage Low (no pull-up)	$V_{IN} = V_{SS}$, V_{DD} = 5.5 V	-10			µA
I_{OZ}	Output Leakage (no pull-up)	$V_{IN} = V_{DD}$ or V_{SS}, V_{DD} = 5.5 V	-10		10	µA
V_{IL}	CMOS Input Low Voltage				0.3 x V_{DD}	V
V_{IH}	CMOS Input High Voltage		0.7 x V_{DD}			V
V_T	CMOS Switching Threshold	V_{DD} = 5.0 V, 25°C		2.4		V
V_{OL}	CMOS Output Low Voltage	I_{OL} = See Table A-5 (sinking) V_{DD} = 4.5 V		0.2	0.4	V
V_{OH}	CMOS Output High Voltage	I_{OH} = See Table A-5 (sourcing) V_{DD} = 4.5 V	0.7 x V_{DD}	4.2		V

A.4 I/O Buffer Sourcing and Sinking Limit

Unless otherwise specified, the Rabbit I/O buffers are capable of sourcing and sinking 8 mA of current per pin at full AC switching speed. Full AC switching assumes a 25.8 MHz CPU clock and capacitive loading on address and data lines of less than 100 pF per pin. Address pin A0 and data pin D0 are rated at 16 mA each. Pins A1–A12 and D1–D7 are each rated at 8 mA. The absolute maximum operating voltage on all I/O is V_{DD} + 0.5 V or 5.5 V.

Table A-5 shows the AC and DC output drive limits of the parallel I/O buffers when the Rabbit 2000 is used in the RCM2200.

Table A-5. I/O Buffer Sourcing and Sinking Capability

Pin Name	Output Drive Sourcing*/Sinking† Limits (mA)	
Output Port Name	Full AC Switching SRC/SNK	Maximum‡ DC Output Drive SRC/SNK
PA [7:0]	8/8	12/12
PB [7:6]	8/8	12/12
PC [6, 2, 0]	8/8	12/12
PD [5:4]	8/8	12/12
PD [3:0]**	16/16	25/25
PE [7, 5, 4, 1, 0]	8/8	12/12

* The maximum DC sourcing current for I/O buffers between V_{DD} pins is 112 mA.

† The maximum DC sinking current for I/O buffers between V_{SS} pins is 150 mA.

‡ The maximum DC output drive on I/O buffers must be adjusted to take into consideration the current demands made my AC switching outputs, capacitive loading on switching outputs, and switching voltage.

The current drawn by all switching and nonswitching I/O must not exceed the limits specified in the first two footnotes.

** The combined sourcing from Port D [7:0] may need to be adjusted so as not to exceed the 112 mA sourcing limit requirement specified in Note 1.

A.5 Jumper Configurations

Figure A-5 shows the header locations used to configure the various RCM2200 options via jumpers.

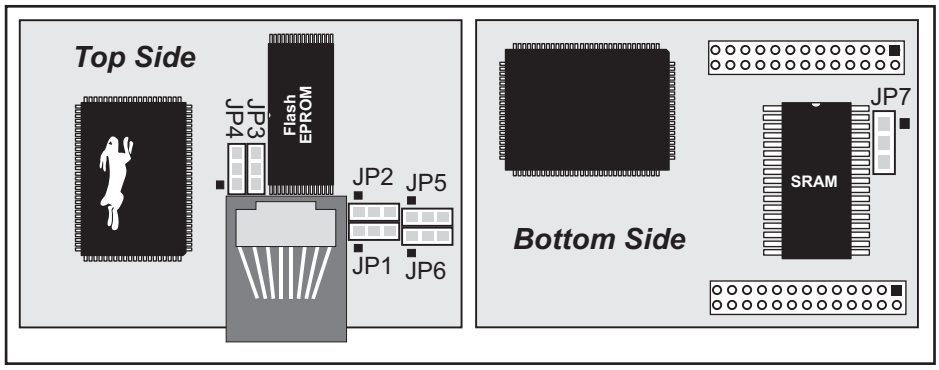


Figure A-5. Location of RCM2200 Configurable Positions

Table A-6 lists the configuration options.

Table A-6. RCM2200 Jumper Configurations

Header	Description	Pins Connected		Factory Default
JP1	Flash Memory Size (U8—RCM2250 only)	1–2	128K/256K	✕
		2–3	512K	
JP2	Flash Memory Bank Select (U8—RCM2250 only)	1–2	Normal Mode	✕
		2–3	Bank Mode	
JP3	Flash Memory Bank Select (U3)	1–2	Normal Mode	✕
		2–3	Bank Mode	
JP4	Flash Memory Size (U3)	1–2	128K/256K	✕
		2–3	512K	
JP5	Flash Memory Bank Select (U7—not installed)	1–2	Normal Mode	—
		2–3	Bank Mode	
JP6	Flash Memory Size (U7—not installed)	1–2	128K/256K	—
		2–3	512K	
JP7	SRAM Size	1–2	128K	RCM2200 RCM2210
		2–3	512K	RCM2250

NOTE: The jumper connections are made using 0 Ω surface-mounted resistors.

A.6 Conformal Coating

The areas around the 32 kHz real-time clock crystal oscillator has had the Dow Corning silicone-based 1-2620 conformal coating applied. The conformally coated area is shown in Figure A-6. The conformal coating protects these high-impedance circuits from the effects of moisture and contaminants over time.

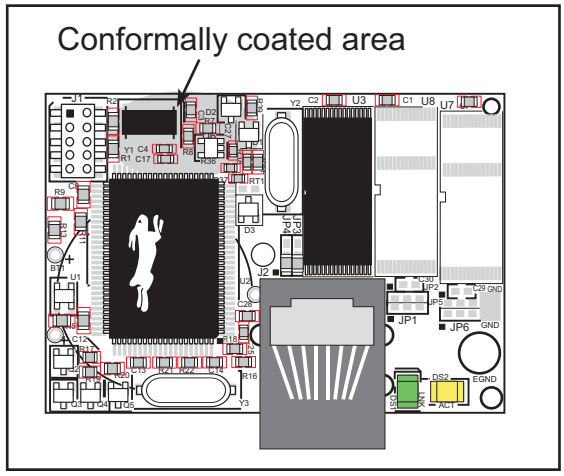


Figure A-6. RCM2200 Areas Receiving Conformal Coating

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 Technical Note TN303, ***Conformal Coatings***.

APPENDIX B. PROTOTYPING BOARD

Appendix B describes the features and accessories of the Prototyping Board, and explains the use of the Prototyping Board to demonstrate the RCM2200 and to build prototypes of your own circuits.

B.1 Prototyping Board

The Prototyping Board included in the Development Kit makes it easy to connect an RCM2200 module to a power supply and a PC workstation for development. It also provides some basic I/O peripherals (switches and LEDs), as well as a prototyping area for more advanced hardware development.

For the most basic level of evaluation and development, the Prototyping Board can be used without modification.

As you progress to more sophisticated experimentation and hardware development, modifications and additions can be made to the board without modifying or damaging the RCM2200 module itself.

The Prototyping Board is shown below in Figure B-1, with its main features identified.

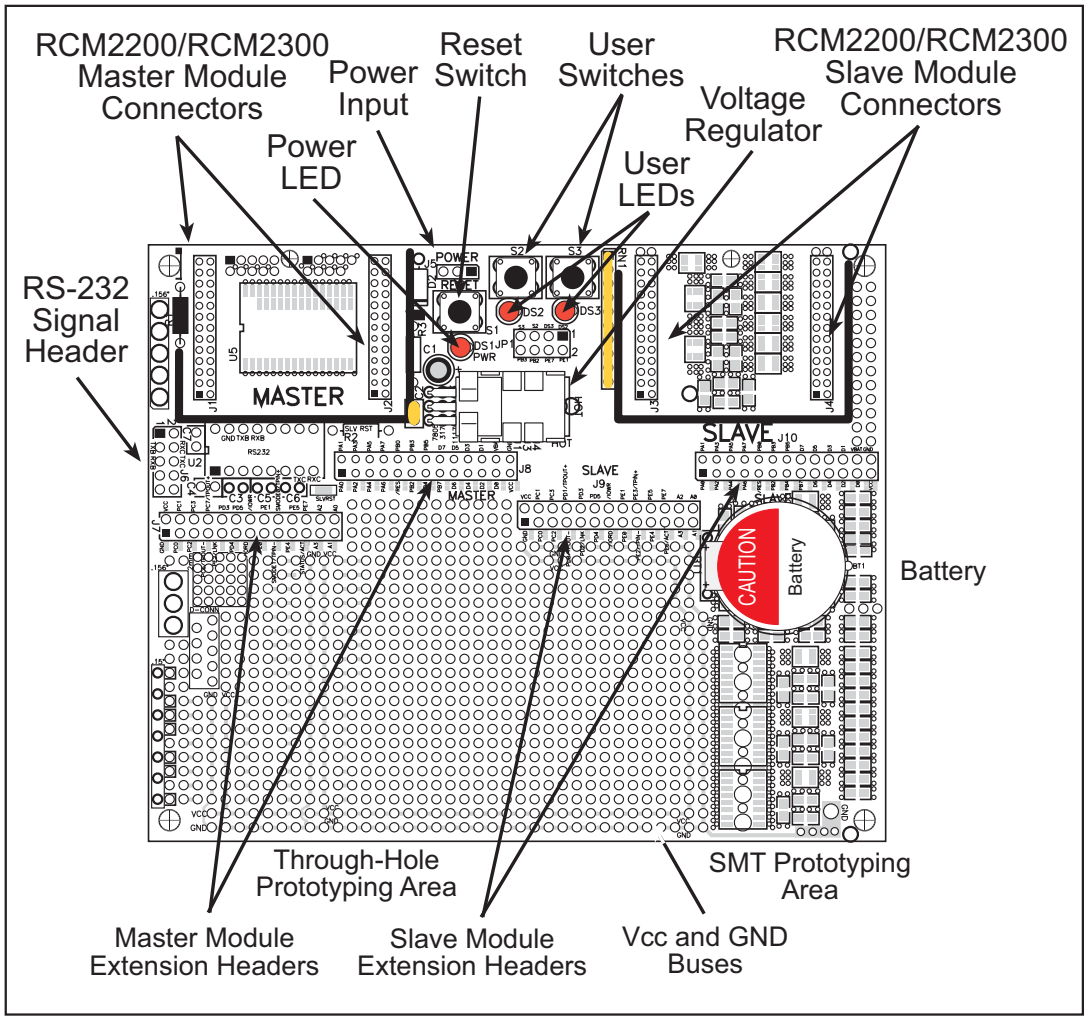


Figure B-1. RCM2200/RCM2300 Prototyping Board

B.1.1 Prototyping Board Features

- **Power Connection**—A 3-pin header is provided for connection to the power supply. Note that it is symmetrical, with both outer pins connected to ground and the center pin connected to the raw V+ input. The cable of the wall transformer provided with the North American version of the development kit ends in a connector that is correctly connected in either orientation.

  Users providing their own power supply should ensure that it delivers 8–24 V DC at not less than 500 mA. The voltage regulator will get warm while in use. (Lower supply voltages will reduce thermal dissipation from the device.)

- **Regulated Power Supply**—The raw DC voltage provided at the POWER IN jack is routed to a 5 V linear voltage regulator, which provides stable power to the RCM2200 module and the Prototyping Board. A Shottky diode protects the power supply against damage from reversed raw power connections.

- **Power LED**—The power LED lights whenever power is connected to the Prototyping Board.

- **Reset Switch**—A momentary-contact, normally open switch is connected directly to the RCM2200's /**RES** pin. Pressing the switch forces a hardware reset of the system.

- **I/O Switches and LEDs**—Two momentary-contact, normally open switches are connected to the PB2 and PB3 pins of the master RCM2200 module and may be read as inputs by sample applications.

  Two LEDs are connected to the PE1 and PE7 pins of the master module, and may be driven as output indicators by sample applications.

  The LEDs and switches are connected through JP1, which has traces shorting adjacent pads together. These traces may be cut to disconnect the LEDs, and an 8-pin header soldered into JP1 to permit their selective reconnection with jumpers. See Figure B-4 for details.

- **Expansion Areas**—The Prototyping Board is provided with several unpopulated areas for expansion of I/O and interfacing capabilities. See the next section for details.

- **Prototyping Area**—A generous prototyping area has been provided for the installation of through-hole components. Vcc (5 V DC) and Ground buses run around the edge of this area. An area for surface-mount devices is provided to the right of the through-hole area. (Note that there are SMT device pads on both top and bottom of the Prototyping Board.) Each SMT pad is connected to a hole designed to accept a 30 AWG solid wire.

- **Slave Module Connectors**—A second set of connectors is pre-wired to permit installation of a second, slave RCM2200 or RCM2300 module. This capability is reserved for future use, although the schematics in this manual contain all of the details an experienced developer will need to implement a master-slave system.

B.1.2 Prototyping Board Expansion

The Prototyping Board comes with several unpopulated areas, which may be filled with components to suit the user's development needs. After you have experimented with the sample programs from the ***RabbitCore RCM2200 User's Manual***, you may wish to expand the board's capabilities for further experimentation and development. Refer to the Prototyping Board schematic (090–0122) for details as necessary.

- **Module Extension Headers**—The complete pin sets of both the Master and Slave RabbitCore modules are duplicated at these two sets of headers. Developers can solder wires directly into the appropriate holes, or, for more flexible development, 26-pin header strips can be soldered into place. See Figure B-5 for the header pinouts.
- **RS-232**—Two 2-wire or one 4-wire RS-232 serial port can be added to the Prototyping Board by installing a driver IC and four capacitors. The Maxim MAX232CPE driver chip or a similar device is recommended for the U2. Refer to the Prototyping Board schematic for additional details.

  A 10-pin 0.1-inch spacing header strip can be installed at J6 to permit connection of a ribbon cable leading to a standard DE-9 serial connector.

  All RS-232 port components mount to the underside of the Prototyping Board, between the Master module connectors.

  NOTE: The RS-232 chip, capacitors and header strip are available from electronics distributors such as Digi-Key.
- **Prototyping Board Component Header**—Four I/O pins from the module are hard-wired to the Prototyping Board LEDs and switches.

B.2 Mechanical Dimensions and Layout

Figure B-2 shows the mechanical dimensions and layout for the RCM2200 Prototyping Board.

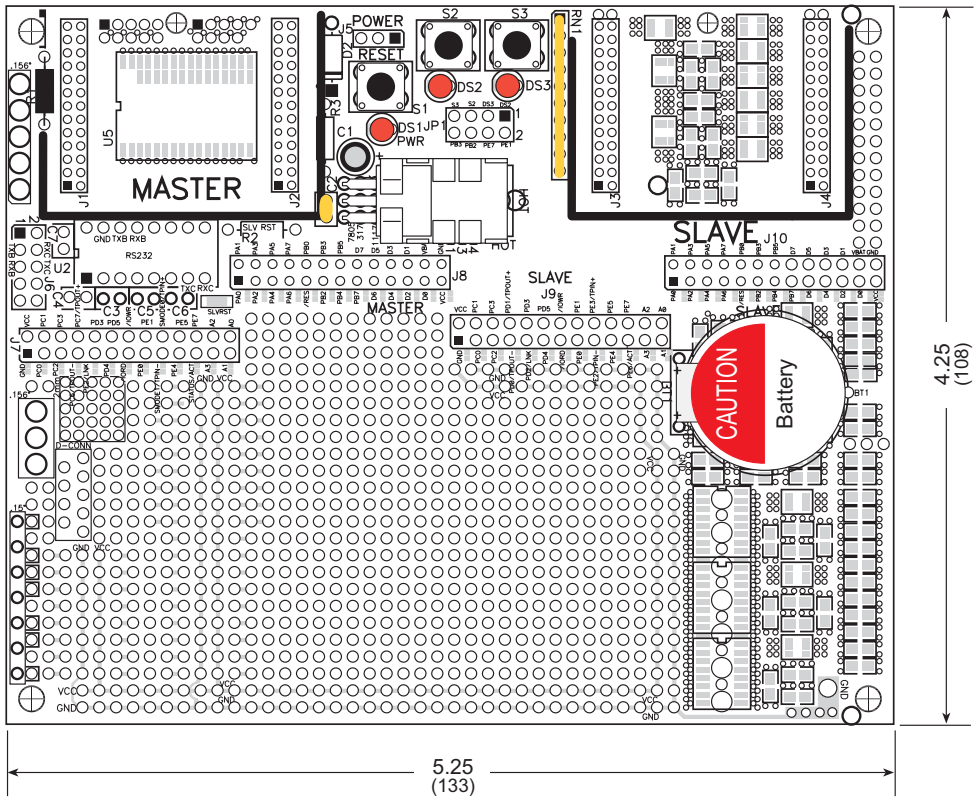


Figure B-2. RCM2200 Prototyping Board Dimensions

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

Table B-1. RCM2200 Prototyping Board Specifications

Parameter	Specification
Board Size	4.25" × 5.25" × 1.00" (108 mm × 133 mm × 25 mm)
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Input Voltage	7.5 V to 25 V DC
Maximum Current Draw (including user-added circuits)	1 A at 12 V and 25°C, 0.7 A at 12 V and 70ºC
Prototyping Area	2.4" × 4.0" (61 mm × 102 mm) throughhole, 0.1" spacing, additional space for SMT components
Standoffs/Spacers	4, accept 6-32 × 3/8 screws

B.3 Power Supply

The RCM2200 requires a regulated 5 V ± 0.25 V DC power source to operate. Depending on the amount of current required by the application, different regulators can be used to supply this voltage.

The Prototyping Board has an onboard 7805 or equivalent linear regulator that is easy to use. Its major drawback is its inefficiency, which is directly proportional to the voltage drop across it. The voltage drop creates heat and wastes power.

A switching power supply may be used in applications where better efficiency is desirable. The LM2575 is an example of an easy-to-use switcher. This part greatly reduces the heat dissipation of the regulator. The drawback in using a switcher is the increased cost.

The Prototyping Board itself is protected against reverse polarity by a Shottky diode at D2 as shown in Figure B-3.

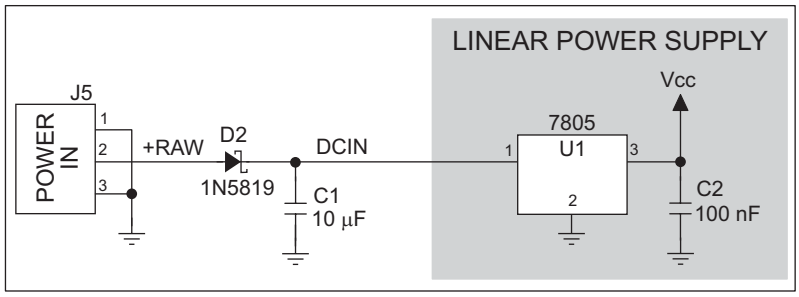


Figure B-3. Prototyping Board Power Supply

B.4 Using the Prototyping Board

The Prototyping Board is actually both a demonstration board and a prototyping board. As a demonstration board, it can be used to demonstrate the functionality of the RCM2200 right out of the box without any modifications to either board. There are no jumpers or dip switches to configure or misconfigure on the Prototyping Board so that the initial setup is very straightforward.

The Prototyping Board comes with the basic components necessary to demonstrate the operation of the RCM2200. Two LEDs (DS2 and DS3) are connected to PE1 and PE7, and two switches (S2 and S3) are connected to PB2 and PB3 to demonstrate the interface to the Rabbit 2000 microprocessor. Reset switch S1 is the hardware reset for the RCM2200.

To maximize the availability of RCM2200 resources, the demonstration hardware (LEDs and switches) on the Prototyping Board may be disconnected. This is done by cutting the traces below the silk-screen outline of header JP1 on the bottom side of the Prototyping Board. Figure B-4 shows the four places where cuts should be made. Cut the traces between the rows as shown. An exacto knife would work nicely to cut the traces. Alternatively, a small standard screwdriver may be carefully and forcefully used to wipe through the PCB traces.Use jumpers across the positions on JP1 if you need to reconnect any of the devices later on.

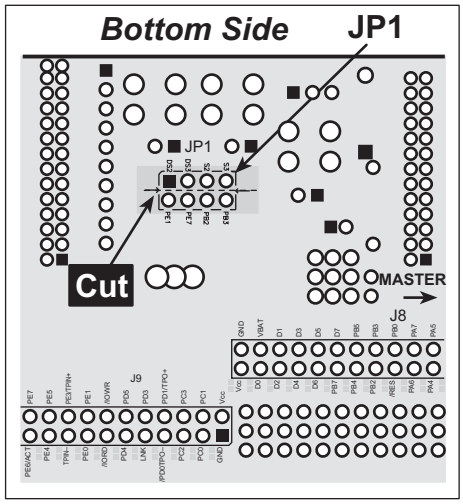


Figure B-4. Where to Cut Traces to Permanently Disable Demonstration Hardware on Prototyping Board

The power LED (PWR) and the RESET switch remain connected. Jumpers across the appropriate pins on header JP1 can be used to reconnect specific demonstration hardware later if needed.

Table B-2. Prototyping Board Jumper Settings

Header JP1	
Pins	**Description**
1–2	PE1 to LED DS2
3–4	PE7 to LED DS3
5–6	PB2 to Switch S2
7–8	PB3 to Switch S3

Note that the pinout at location JP1 on the bottom side of the Prototyping Board (shown in Figure B-4) is a mirror image of the top-side pinout.

The Prototyping Board provides the user with RCM2200 connection points brought out conveniently to labeled points at headers J7 and J8 on the Prototyping Board. Small to medium

circuits can be prototyped using point-to-point wiring with 20 to 30 AWG wire between the prototyping area and the holes at locations J7 and J8. The holes are spaced at 0.1" (2.5 mm), and 40-pin headers or sockets may be installed at J7 and J8. The pinouts for locations J7 and J8, which correspond to headers J1 and J2, are shown in Figure B-5.

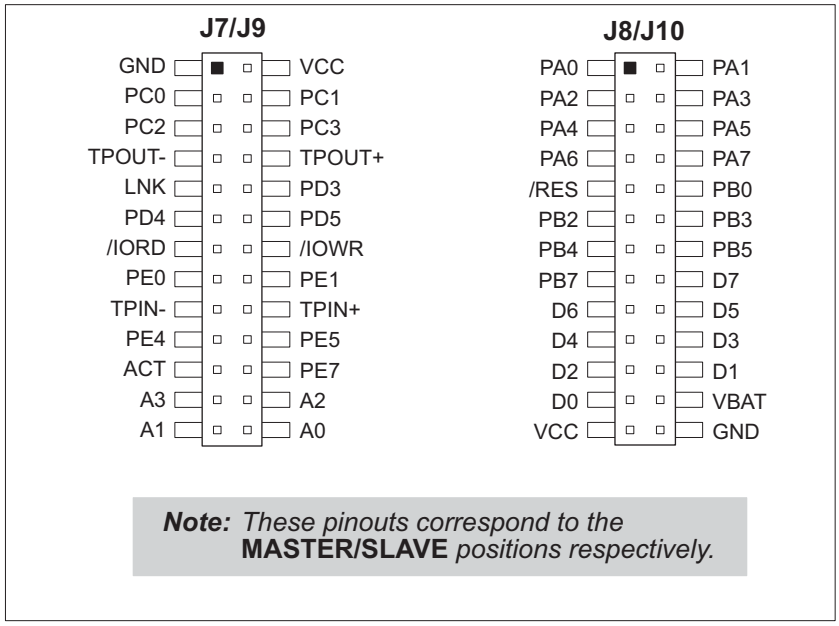


Figure B-5. RCM2200 Prototyping Board Pinout (Top View)

The small holes are also provided for surface-mounted components that may be installed to the right of the prototyping area.

There is a 2.4" × 4" through-hole prototyping space available on the Prototyping Board. VCC and GND traces run along the edge of the Prototyping Board for easy access. A GND pad is also provided at the lower right for alligator clips or probes.

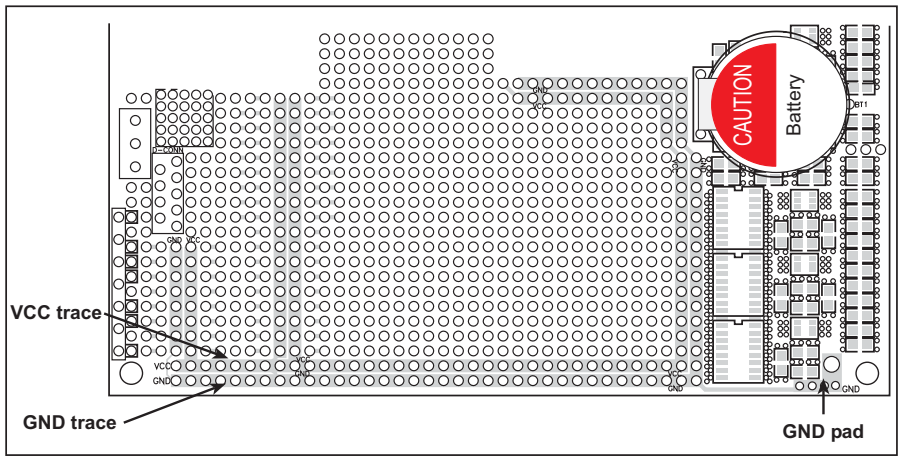


Figure B-6. VCC and GND Traces Along Edge of Prototyping Board

B.4.1 Adding Other Components

There is room on the Prototyping Board for a user-supplied RS-232 transceiver chip at location U2 and a 10-pin header for serial interfacing to external devices at location J6. A Maxim MAX232 transceiver is recommended. When adding the MAX232 transceiver at position U2, you must also add 100 nF charge storage capacitors at positions C3–C7 as shown in Figure B-7.

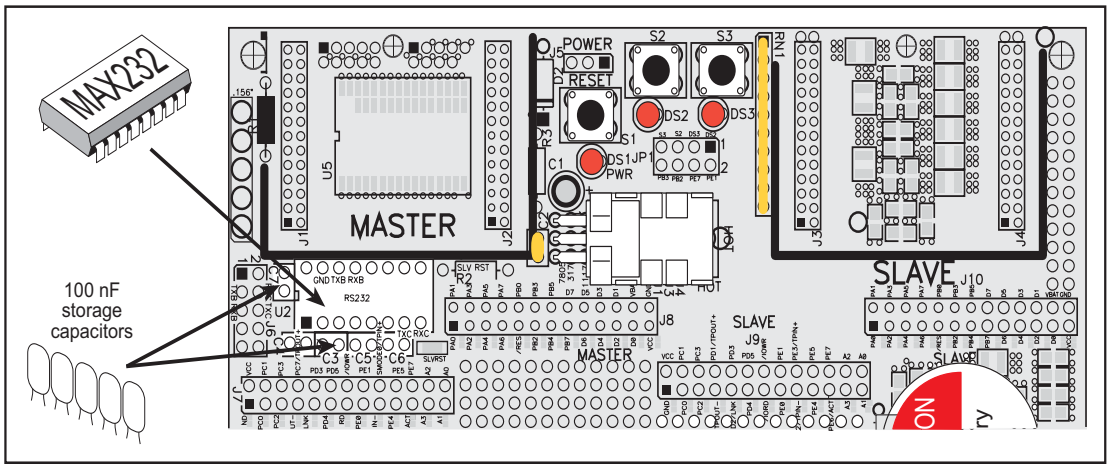


Figure B-7. Location for User-Supplied RS-232 Transceiver and Charge Storage Capacitors on Back Side of Prototyping Board

NOTE: The board that is supplied with the DeviceMate Development Kit already has the RS-232 chip and the storage capacitors installed, and is called the DeviceMate Demonstration Board.

There are two sets of pads that can be used for surface mount prototyping SOIC devices. The silk screen layout separates the rows into six 16-pin devices (three on each side). However, there are pads between the silk screen layouts giving the user two 52-pin (2×26) SOIC layouts with 50 mil pin spacing. There are six sets of pads that can be used for 3- to 6-pin SOT23 packages. There are also 60 sets of pads that can be used for SMT resistors and capacitors in an 0805 SMT package. Each component has every one of its pin pads connected to a hole in which a 30 AWG wire can be soldered (standard wire wrap wire can be soldered in for point-to-point wiring on the Prototyping Board). Because the traces are very thin, carefully determine which set of holes is connected to which surface-mount pad.

There is also a space above the space for the RS-232 transceiver that can accommodate a large surface-mounted SOIC component.

APPENDIX C. POWER SUPPLY

Appendix C provides information on the current requirements of the RCM2200, and includes some background on the chip select circuit used in power management.

C.1 Power Supplies

The RCM2200 requires a regulated 5 V ± 0.25 V DC power source. The RabbitCore design presumes that the voltage regulator is on the user board, and that the power is made available to the RabbitCore board through headers J4 and J5.

An RCM2200 with no loading at the outputs operating at 22.1 MHz typically draws 134 mA. The RCM2200 will consume an additional 10 mA when the programming cable is used to connect the programming header, J1, to a PC.

C.1.1 Battery-Backup Circuits

The RCM2200 does not have a battery, but there is provision for a customer-supplied battery to back up SRAM and keep the internal Rabbit 2000 real-time clock running.

Header J5, shown in Figure C-1, allows access to the external battery. This header makes it possible to connect an external 3 V power supply. This allows the SRAM and the internal Rabbit 2000 real-time clock to retain data with the RCM2200 powered down.

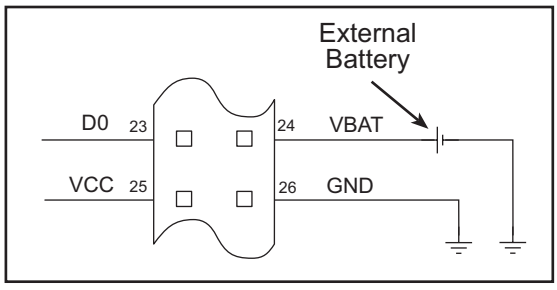


Figure C-1. External Battery Connections at Header J5

A lithium battery with a nominal voltage of 3 V and a minimum capacity of 165 mA·h is recommended. A lithium battery is strongly recommended because of its nearly constant nominal voltage over most of its life.

The drain on the battery by the RCM2200 is typically 16 µA when no other power is supplied. If a 950 mA·h battery is used, the battery can last more than 6 years:

$$\frac{950\ \text{mA·h}}{16\ \mu\text{A}} = 6.8\ \text{years.}$$

The actual life in your application will depend on the current drawn by components not on the RCM2200 and the storage capacity of the battery. Note that the shelf life of a lithium ion battery is ultimately 10 years. The RCM2200 does not drain the battery while it is powered up normally.

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 U6, 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 RCM2200.

Figure C-2 shows the RCM2200 battery-backup circuit.

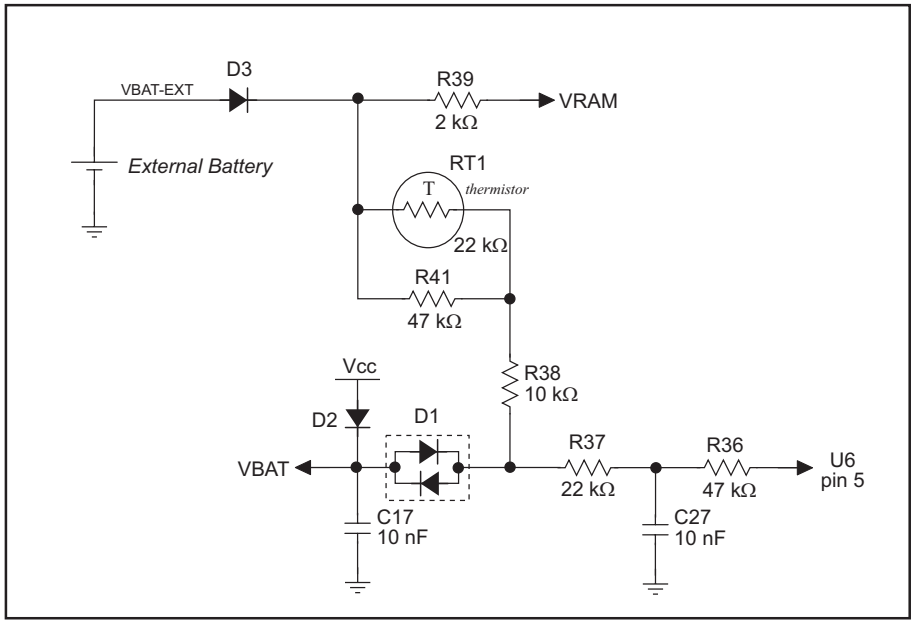


Figure C-2. RCM2200 Battery-Backup Circuit

C.1.2 Reset Generator

The RCM2200 uses a reset generator, U1, 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. The RCM2200 has a reset output, pin 9 on header J5.

C.2 Chip Select Circuit

The RCM2200 has provision for battery backup, which kicks in to keep VRAM from dropping below 2 V.

When the RCM2200 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 Vcc 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 switch accomplishes this task for the CS signal line.

Figure C-3 shows a schematic of the chip select control switch.

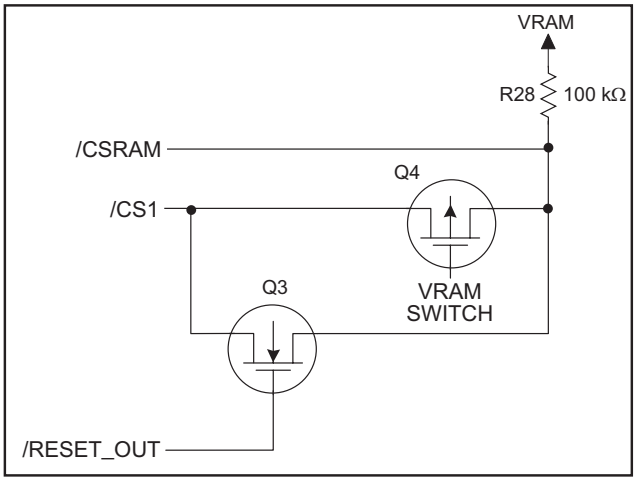


Figure C-3. Chip Select Control Switch

In a powered-up condition, the CS control switch 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 as high as the battery voltage). Q3 and Q4 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 (R28). 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 Q3 and Q4 are of opposite polarity so that a rail-to-rail voltages can be passed. When the /CS1 voltage is low, Q3 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.

APPENDIX D. SAMPLE CIRCUITS

This appendix details several basic sample circuits that can be used with the RCM2200 modules.

- RS-232/RS-485 Serial Communication
- Keypad and LCD Connections
- External Memory
- D/A Converter

D.1 RS-232/RS-485 Serial Communication

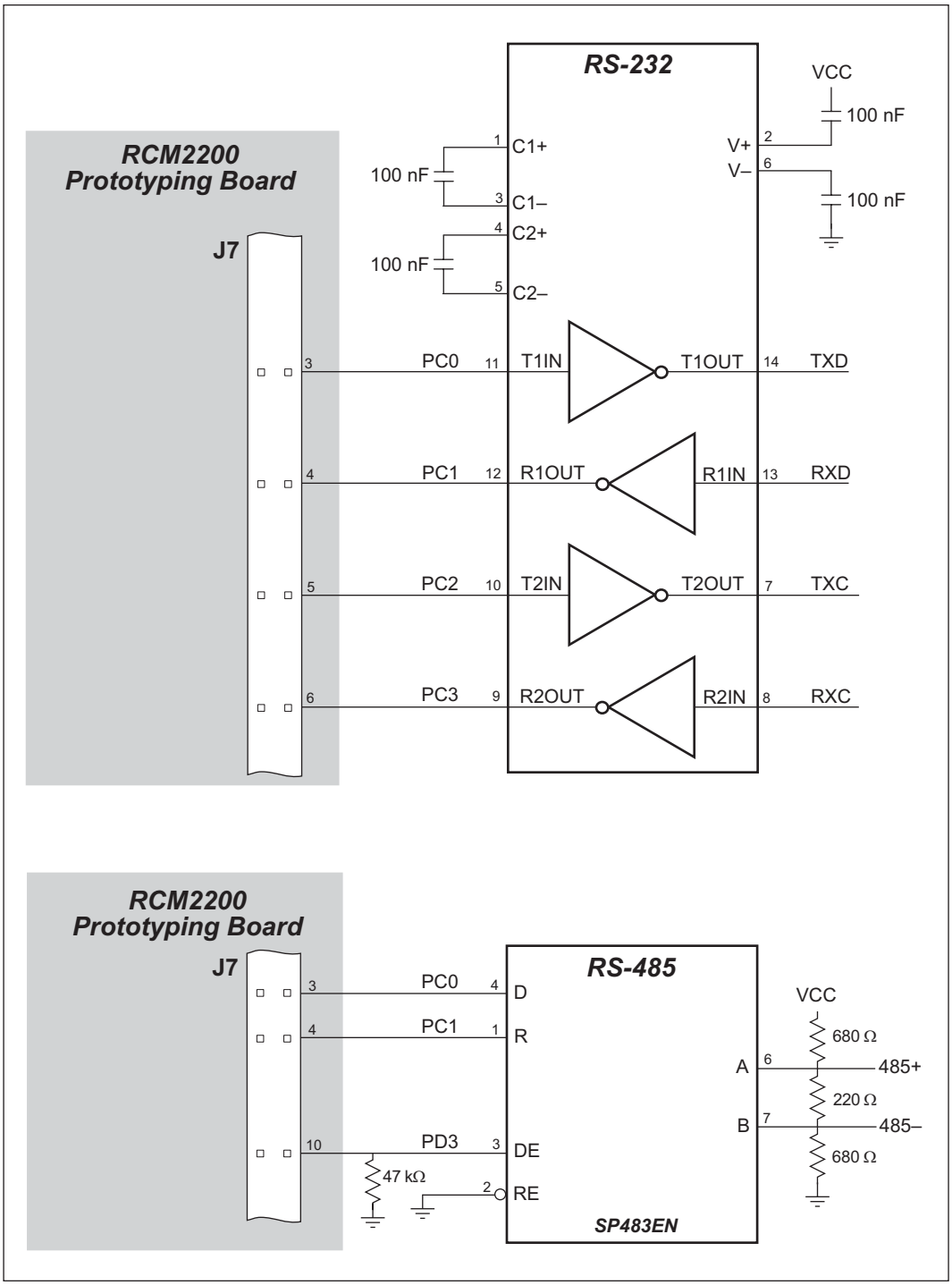


Figure D-1. Sample RS-232 and RS-485 Circuits

Sample Program: **PUTS.C** in **SAMPLES\RCM2200**.

D.2 Keypad and LCD Connections

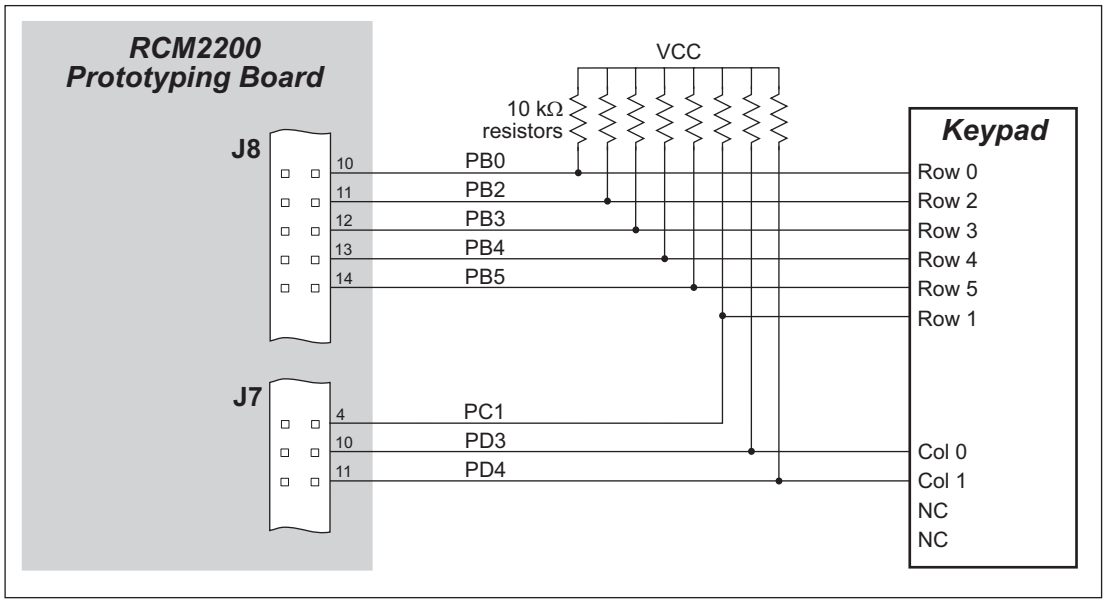


Figure D-2. Sample Keypad Connections

Sample Program: **KEYLCD.C** in **SAMPLES\RCM2200**.

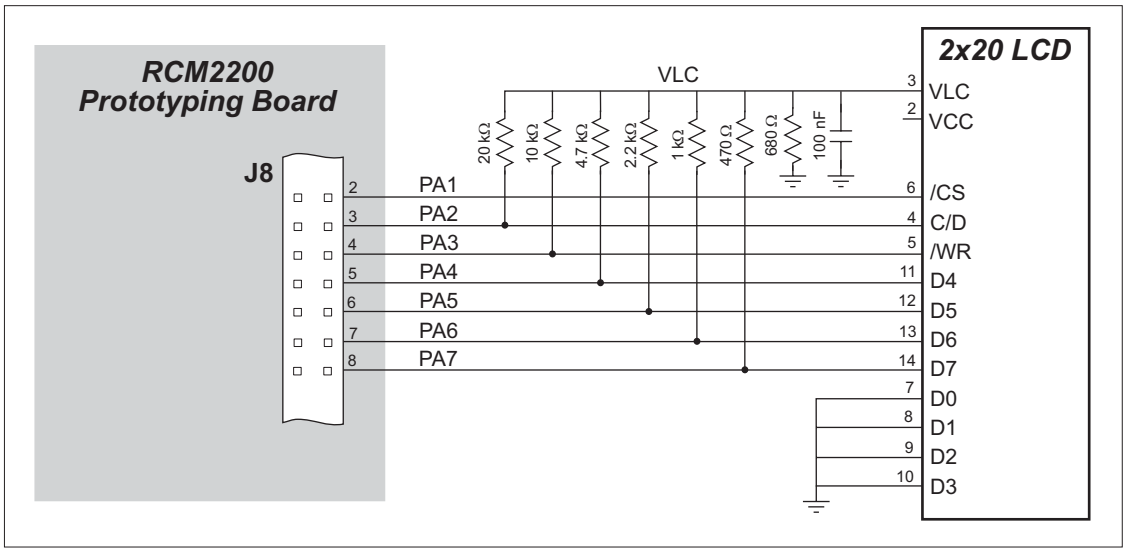


Figure D-3. Sample LCD Connections

Sample Program: **KEYLCD.C** in **SAMPLES\RCM2200**.

D.3 External Memory

The sample circuit can be used with an external 64K memory device. Larger SRAMs can be written to using this scheme by using other available Rabbit 2000 ports (parallel ports A to E) as address lines.

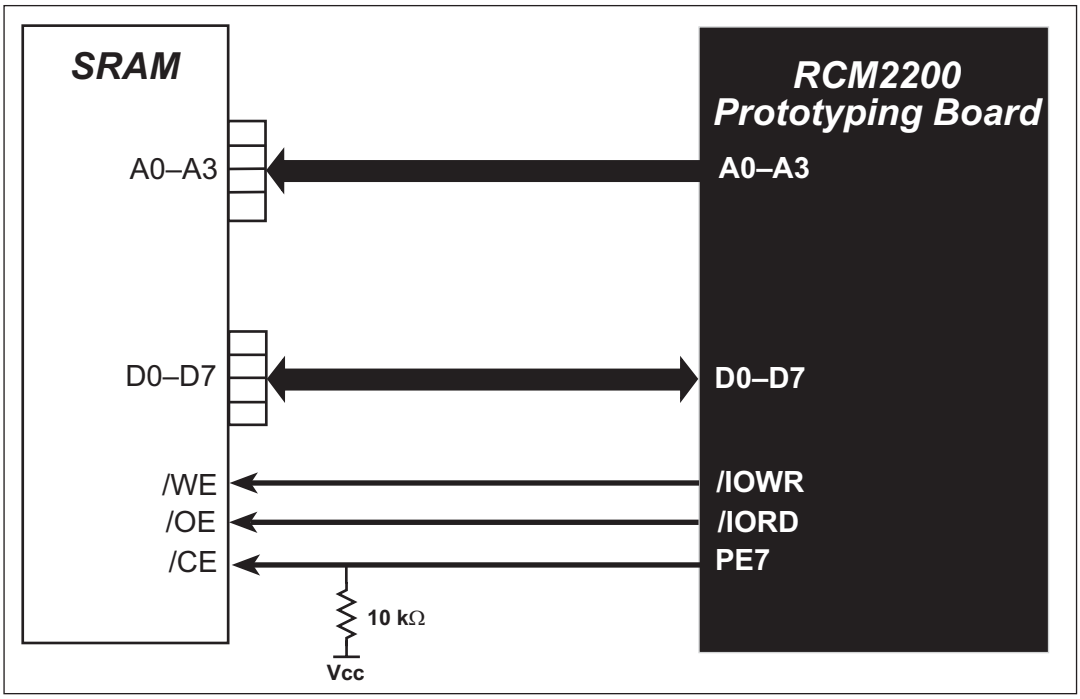


Figure D-4. Sample External Memory Connections

Sample Program: **EXTSRAM.C** in **SAMPLES\RCM2200**.

D.4 D/A Converter

The output will initially be 0 V to -10.05 V after the first inverting op-amp, and 0 V to +10.05 V after the second inverting op-amp. All lows produce 0 V out, FF produces 10 V out. The output can be scaled by changing the feedback resistors on the op-amps. For example, changing 5.11 kΩ to 2.5 kΩ will produce an output from 0 V to -5 V. Op-amps with a very low input offset voltage are recommended.

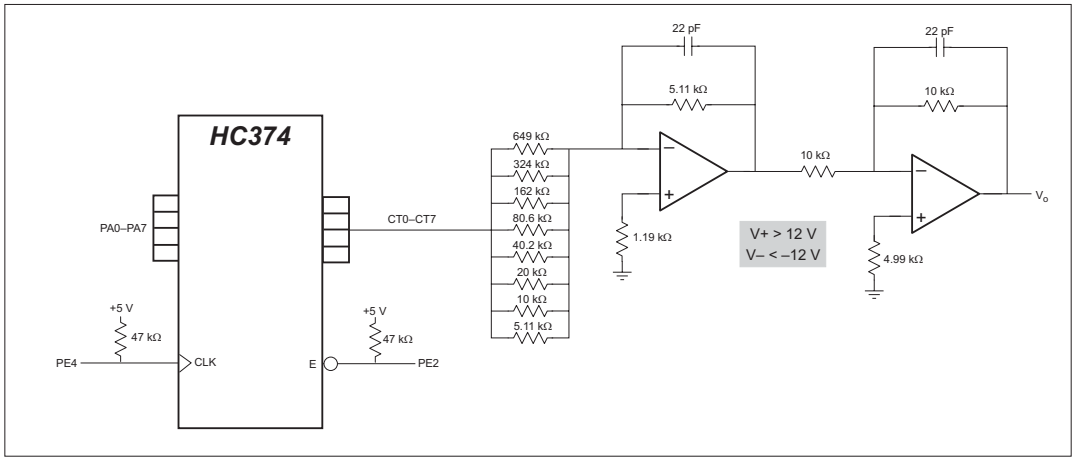


Figure D-5. Sample D/A Converter Connections

INDEX

A

additional information
  online documentation 4

B

backup-battery circuit
  external battery connections . 71
battery life 72
battery-backup circuit
  reset generator 72
bus loading 54

C

clock doubler 28
conformal coating 59

D

Development Kit 3
  DeviceMate 69
  RCM2200 3
digital I/O 19
  I/O buffer sourcing and sinking limits 57
  memory interface 20
  SMODE0 24
  SMODE1 24
digital inputs 20
digital outputs 20
dimensions
  Prototyping Board 65
  RCM2200 50
Dynamic C 3, 31
  add-on modules 35
  Rabbit Embedded Security Pack 35
  sample programs 11
  standard features 32
  debugging 32
  telephone-based technical support 35
  upgrades and patches 35
  USB port settings 9

E

EMI
  spectrum spreader feature . 29
Ethernet cables 37
Ethernet connections 37, 40
  10Base-T 40
  10Base-T Ethernet card 37
  additional resources 48
  Ethernet cables 40
  Ethernet hub 37
  IP addresses 39, 41
  steps 37, 38
Ethernet port 23
  pinout 23
exclusion zone 51
external interrupts 33

F

features 1, 2
  Prototyping Board 62, 63
flash memory
  using second 256K flash memory 31
flash memory addresses
  user blocks 27

H

hardware connections
  install RCM2200 on Prototyping Board 6
  power supply 8
  programming cable 7
hardware reset 8

I

I/O buffer sourcing and sinking limits 57
IP addresses 41
  how to set 45
  how to set PC IP address ... 46

J

JP1
  Prototyping Board 67
jumper configurations 58
  JP1 (U8 flash memory size) 58
  JP2 (U8 flash memory bank select) 27, 58
  JP3 (U3 flash memory bank select) 27, 58
  JP4 (U3 flash memory size) 58
  JP5 (U7 flash memory bank select) 27, 58
  JP6 (U7 flash memory size) 58
  JP7 (SRAM size) 58
  jumper locations 58

M

manuals 4

P

PCLK output 33
physical mounting 53
pinout
  Ethernet port 23
  Prototyping Board 68
  RCM2200 20
  alternate configurations 21, 22
power supplies 71
  chip select circuit 71
power supply
  connections 8
Program Mode 25
  switching modes 25
programming cable
  PROG connector 25
  RCM2200 connections 7
programming port 24

Prototyping Board 62
adding RS-232 transceiver 69
dimensions 65
expansion area 64
features 62, 63
header JP1 location 67
mounting RCM2200 6
optional connections to Rabbit 2000 parallel ports 67
optional header JP1 67
pinout 68
power supply 66
power supply connections ...8
prototyping area 68
specifications 65
Vcc and GND traces 68

R

Rabbit subsystems 19
RCM2200
mounting on Prototyping Board 6
reset 8
Run Mode 25
switching modes 25
running TCP/IP sample programs 39

S

sample circuits 75
D/A converter 79
external memory 78
sample programs 11
getting to know the RCM2200
EXTSRAM.C 12
FLASHLED.C 12, 16
FLASHLEDS.C 12, 17
KEYLCD.C 13
TOGGLELED.C 12, 18
how to set IP address 45
PONG.C 9
RCM2200 11
serial communication
MASTER.C 15
PUTS.C 14
SLAVE.C 15
TCP/IP 39
CONSOLE.C 47
ETHCORE1.C 47
MYECHO.C 47
PINGME.C 47
running TCP/IP sample programs 39
SERDCLIENT.C 47
SPCLIENT.C 47
TCPIP 11
serial communication 23
serial ports 23
Ethernet port 23
programming port 24
software
I/O drivers 33
libraries
PACKET.LIB 34
RS232.LIB 34
TCP/IP 34
macros
USE_2NDFLASH_CODE 31
PCLK output 33
sample programs 11
serial communication drivers 34
TCP/IP drivers 34
using second 256K flash memory 31
specifications 49
bus loading 54
digital I/O buffer sourcing and sinking limits 57
dimensions 50
electrical, mechanical, and environmental 52
exclusion zone 51
header footprint 53
headers 53
physical mounting 53
Prototyping Board 65
Rabbit 2000 DC characteristics 56
Rabbit 2000 timing diagram 55
relative pin 1 locations 53
spectrum spreader 29
subsystems
digital inputs and outputs ..19
switching modes 25

T

technical support 10
troubleshooting
changing COM port 9
connections 9

U

USB/serial port converter 7
Dynamic C settings 9
user block
function calls
readUserBlock 27
writeUserBlock 27

SCHEMATICS

090-0120 RCM2200 Schematic

www.rabbit.com/documentation/schemat/090-0120.pdf

090-0122 RCM2200 Prototyping Board Schematic

www.rabbit.com/documentation/schemat/090-0122.pdf

090-0128 Programming Cable Schematic

www.rabbit.com/documentation/schemat/090-0128.pdf

You may use the URL information provided above to access the latest schematics directly.