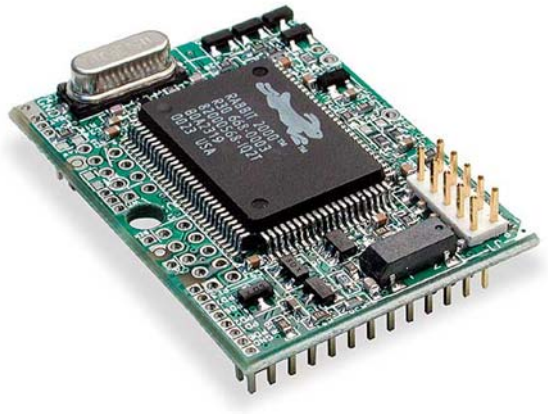
PRODUCT MANUAL



RabbitCore RCM2300

C-Programmable Module

User's Manual

019–0099 • 070831–G

RabbitCore RCM2300 User's Manual

Part Number 019-0099 • 070831–G • Printed in U.S.A.

Trademarks

Rabbit and Dynamic C are registered trademarks of Rabbit Semiconductor Inc.

Rabbit 2000 and RabbitCore are trademarks of Rabbit Semiconductor Inc.

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

Rabbit Semiconductor Inc.

www.rabbit.com

TABLE OF CONTENTS

1. INTRODUCTION

The RabbitCore RCM2300 is a very small advanced core module that incorporates the powerful Rabbit® 2000 microprocessor, flash memory, static RAM, and digital I/O ports, all on a PCB that is just 1.15" × 1.60" (29.2 mm × 40.6 mm).

The RCM2300 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 RCM2300 receives its +5 V power from the user board on which it is mounted. The RabbitCore RCM2300 can interface with all kinds of CMOS-compatible digital devices through the user board.

1.1 RabbitCore RCM2300 Features

- Small size: 1.15" × 1.60" × 0.55"
  (29 mm × 41 mm × 14 mm)
- Microprocessor: Rabbit 2000 running at 22.1 MHz
- 29 parallel I/O lines: 17 configurable for input or output, 8 fixed inputs, 4 fixed outputs
- 11 additional I/O are available via less convenient 0.03" diameter through-hole connection points
- 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 one 10-bit timer with two match registers
- 256K flash memory, 128K SRAM
- Real-time clock
- Watchdog supervisor

- Provision for customer-supplied backup battery either onboard or via header connections
- Four CMOS-compatible serial ports. All the serial ports can be configured asynchronously, and two serial ports can be configured synchronously if so desired. The maximum asynchronous baud rate is 691,200 bps (Dynamic C drivers are capable of handling up to the sustained rate of 345,600 bps), and the maximum synchronous baud rate is 5.5296 Mbps (user-written drivers can sustain a rate of 2.7648 Mbps). One synchronous port clock line is available only on the programming header.
- The programming port is also routed to the 26-pin headers, which allows the user board the ability to reprogram the RCM2300.

Appendix A, "RabbitCore RCM2300 Specifications," provides detailed specifications for the RCM2300.

1.2 Advantages of the RabbitCore RCM2300

- 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.
- Very small size.

1.3 Development and Evaluation Tools

A complete Development Kit, which includes a Prototyping Board and Dynamic C development software, is available for the RCM2300. 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 RCM2300 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 RCM2300 requires Dynamic C v7.04 or later for development. A compatible version is included on the Development Kit CD-ROM.

1.3.2 Development Kit Contents

The RCM2300 Development Kit contains the following items:

- RCM2300 module with 256K flash memory and 128K SRAM.
- RCM2200/RCM2300 Prototyping Board.
- Wall transformer power supply, 12 V DC, 1 A. The power supply is included only with Development Kits sold for the North American market. Overseas users should use a locally available power supply capable of delivering 7.5 V to 25 V DC to the Prototyping Board.
- Programming cable with integrated level-matching circuitry.
- ***Dynamic C*** CD-ROM, with complete product documentation on CD.
- ***Getting Started*** instructions.
- ***Rabbit 2000 Processor Easy Reference*** poster.
- Registration card.

1.4 How to Use This Manual

This user's manual is intended to give users detailed information on the RCM2300 module. It does not contain detailed information on the Dynamic C development environment. Most users will want more detailed information on some or all of these topics in order to put the RCM2300 module to effective use.

1.4.1 Additional Reference Information

In addition to the product-specific information contained in the ***RabbitCore RCM2300 User's Manual*** (this manual), two 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 RCM2300 modules:

- ***Dynamic C User's Manual***
- ***Dynamic C Function Reference Manual***
- ***Rabbit 2000 Microprocessor User's Manual***

1.4.2 Using Online Documentation

We provide the bulk of our user and reference documentation in two electronic formats, HTML and Adobe PDF. We do this for several reasons.

We believe that providing all users with our complete library of product and reference manuals is a useful convenience. However, printed manuals are expensive to print, stock, and ship. Rather than include and charge for manuals that every user may not want, or provide only product-specific manuals, we choose to provide our complete documentation and reference library in electronic form with every Development Kit and with our Dynamic C development environment.

> **NOTE:** The most current version of Adobe Acrobat Reader can always be downloaded from Adobe's Web site at **http://www.adobe.com**. We recommend that you use version 4.0 or later.

Providing this documentation in electronic form saves an enormous amount of paper by not printing copies of manuals that users don't need. It reduces the number of outdated manuals we have to discard from stock as well, and it makes providing a complete library of manuals an almost cost-free option. For one-time or infrequent reference, electronic documents are more convenient than printed ones.

1.4.2.1 Finding Online Documents

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.

1.4.2.2 Printing Electronic Manuals

We recognize that many users prefer printed manuals for some uses. Users can easily print all or parts of those manuals provided in electronic form. The following guidelines may be helpful:

- Print from the Adobe PDF versions of the files, not the HTML versions.
- Print only the sections you will need to refer to more than once.
- Print manuals overnight, when appropriate, to keep from tying up shared resources during the work day.
- If your printer supports duplex printing, print pages double-sided to save paper and increase convenience.
- If you do not have a suitable printer or do not want to print the manual yourself, most retail copy shops (e.g., Kinkos, AlphaGraphics, CopyMax) will print the manual from the PDF file and bind it for a reasonable charge—about what we would have to charge for a printed and bound manual.

2. GETTING STARTED

This chapter describes the RCM2300 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 RabbitCore RCM2300 Development Kit. If you purchased an RCM2300 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 three steps to connecting the Prototyping Board for use with Dynamic C and the sample programs:

1. Attach the RCM2300 to the Prototyping Board.
2. Connect the programming cable between the RCM2300 and the PC.
3. Connect the power supply to the Prototyping Board.

2.1.1 Attach RCM2300 to Prototyping Board

Turn the RCM2300 module so that the header pins and the mounting hole of the RCM2300 line up with the sockets and mounting hole on the Prototyping Board as shown in Figure 1. Align the module header pins from headers J4 and J5 on the bottom side of the RCM2300 into header sockets J1 and J2 on the Prototyping Board.

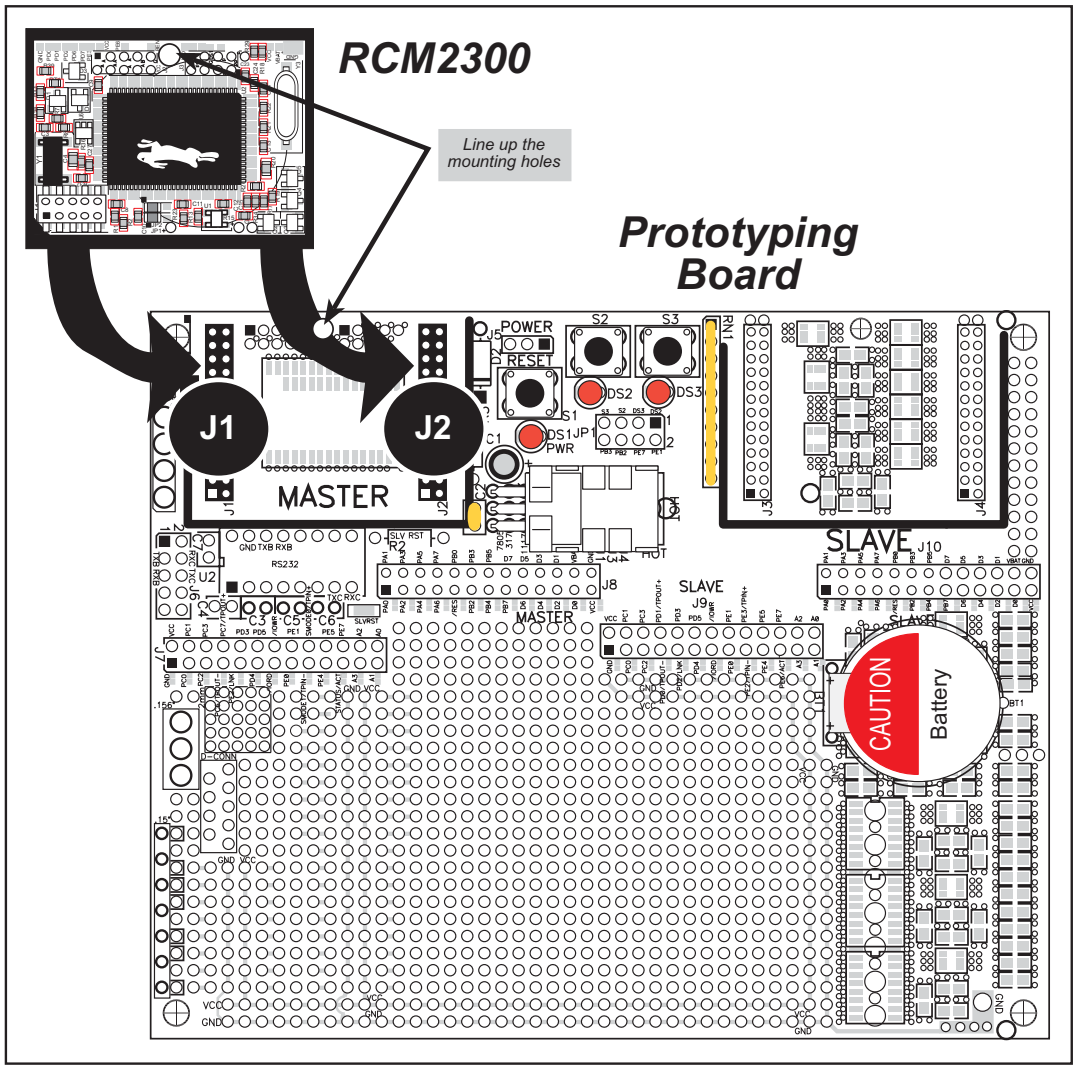


Figure 1. Install the RCM2300 on the Prototyping Board

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 a single module in the **MASTER** position.

NOTE: It is important that you line up the pins on headers J4 and J5 of the RCM2300 exactly with the corresponding pins of headers 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 RCM2300 module to the PC workstation running Dynamic C to permit download of programs and monitoring for debugging.

Connect the 10-pin connector of the programming cable labeled **PROG** to header J1 on the RabbitCore RCM2300 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.

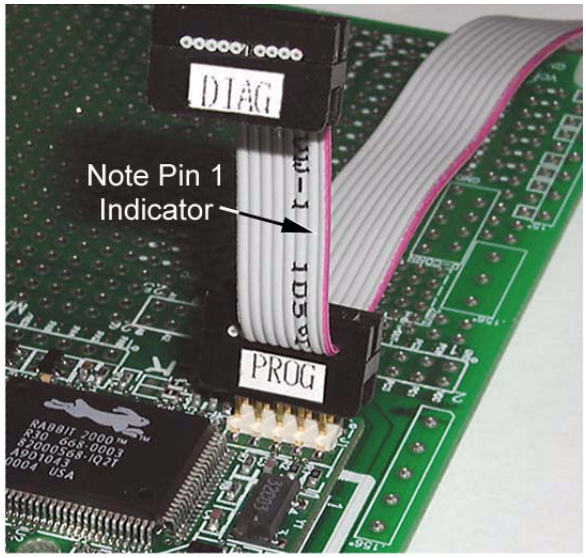


Figure 2. Connect Programming Cable to RCM2300

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. 540-0070) with the programming cable supplied with the RCM2300 Development Kit. Note that not all RS-232/USB converters work with Dynamic C.

2.1.3 Connect Power Supply

When the above connections have been made, you can connect power to the RabbitCore Prototyping Board.

Hook the connector from the wall transformer to header J5 on the Prototyping Board as shown in Figure 3. The connector may be attached either way as long as it is not offset to one side.

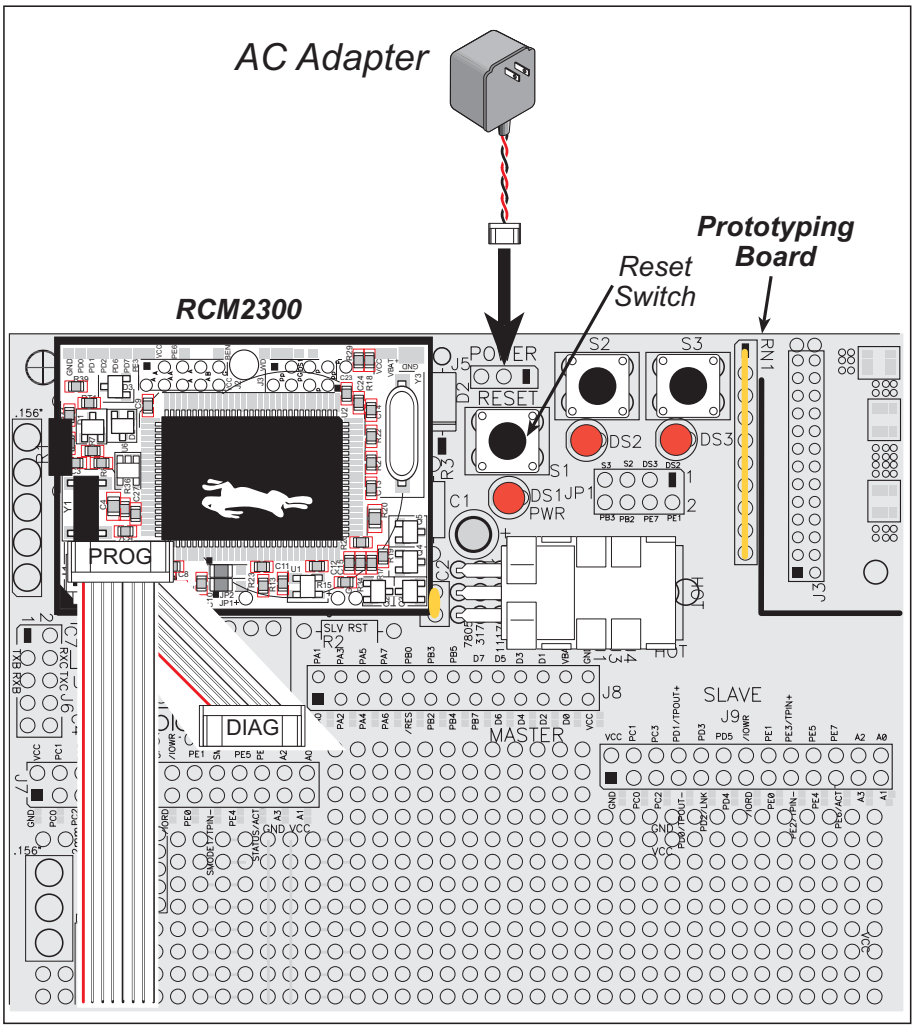


Figure 3. Power Supply Connections

Plug in the wall transformer. The power LED (DS1) on the Prototyping Board should light up. The RCM2300 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 RCM2300 from the board.

2.2 Run a Sample Program

If you already have Dynamic C installed, you are now ready to test your programming connections by running a sample program.

If you are using a USB port to connect your computer to the RCM2300 module, choose **Options > Project Options** and select "Use USB to Serial Converter" under the **Communications** tab.

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 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.

If there are any other problems:

- Check to make sure you are using the **PROG** connector, not the **DIAG** connector, on the programming cable.
- Check both ends of the programming cable to ensure that they are firmly plugged into the PC and the programming port on the RCM2300.
- Ensure that the RCM2300 module is firmly and correctly installed in its connectors on the Prototyping Board.
- 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 active COM port.

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 RCM2300's capabilities.
2. For further development, refer to the ***RabbitCore RCM2300 User's Manual*** for details of the RCM2300'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***, also in the online documentation set.

2.3.1 Technical Support

NOTE: If you purchased your RCM2300 through a distributor or through a Rabbit Semiconductor 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 Semiconductor Technical Bulletin Board at www.rabbit.com/support/bb/.
- Use the Technical Support e-mail form at www.rabbit.com/support/.

3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM2300 (and for all other Rabbit Semiconductor hardware), you must install and use Dynamic C. This chapter provides a tour of the sample programs for the RCM2300 module.

3.1 Sample Programs

To help familiarize you with the RCM2300 modules, Dynamic C includes several sample programs. Loading, executing and studying these programs will give you a solid hands-on overview of the RCM2300'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 RCM2300 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. The sample programs in the Dynamic C **SAMPLES/RCM2300** directory demonstrate the basic operation of the RCM2300.

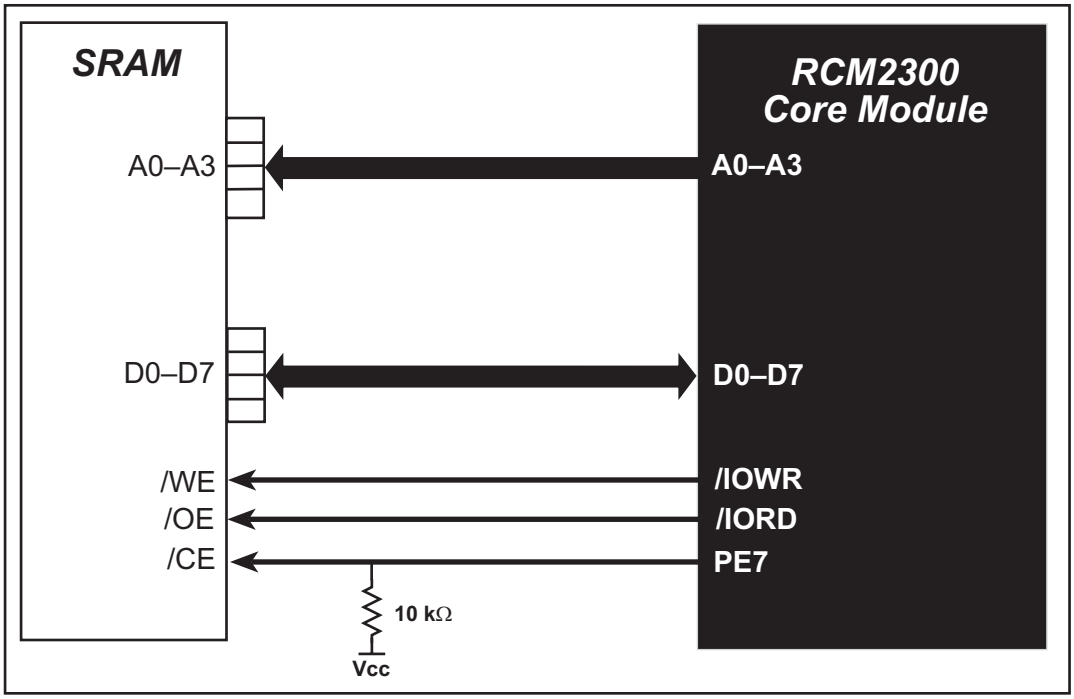
Complete information on Dynamic C is provided in the ***Dynamic C User's Manual***.

3.1.1 Getting to Know the RCM2300

The following sample programs can be found in the **SAMPLES\RCM2300** 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.

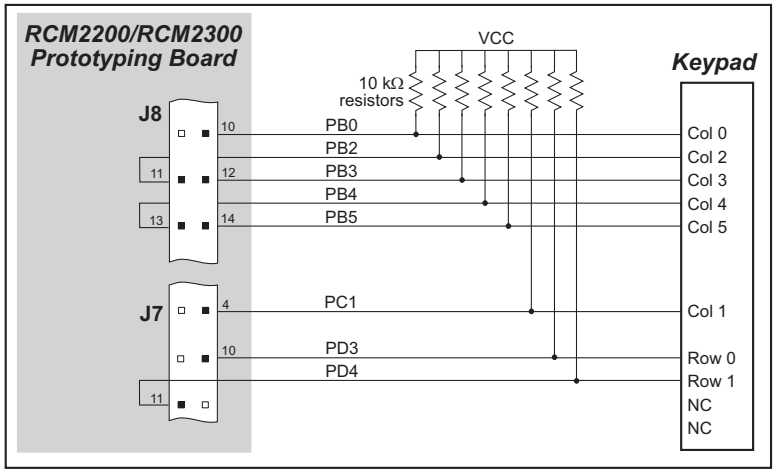


- **FLASHLED.C**—repeatedly flashes LED DS3 on the Prototyping Board on and off. LED DS3 is controlled by Parallel Port E bit 7 (PE7). LED DS2 will remain on continuously.
- **FLASHLEDS.C**—demonstrates the use of coding with assembly instructions, cofunctions, and costatements to flash LEDs DS2 and 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 compile 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.

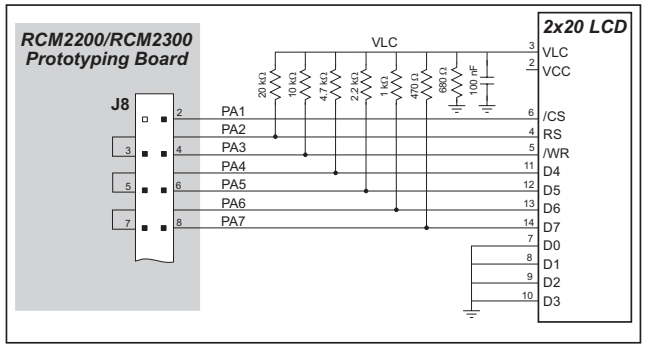
  Connect the keypad to Parallel Ports B, C, and D.

  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

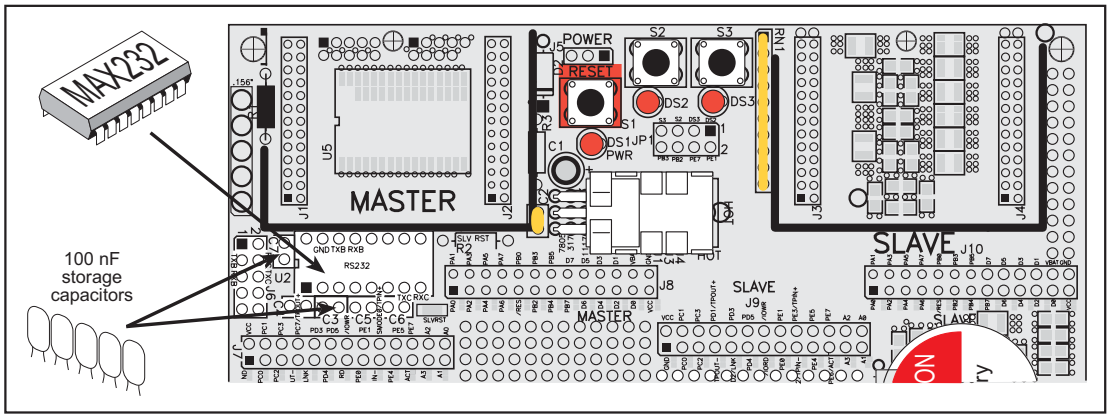


  Once the connections have been made and the sample program isrunning, the LCD will display two rows of 6 dots, each dot representing the corresponding key. When a key is pressed, the corresponding dot will become an asterisk.

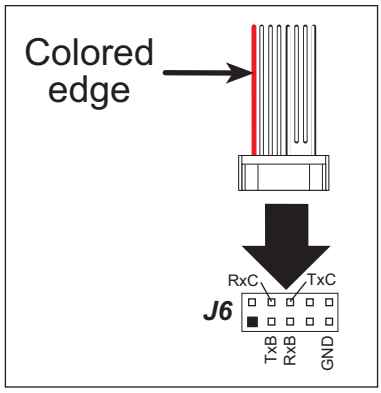
3.1.2 Serial Communication

The following sample programs can be found in the **SAMPLES\RCM2300** 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 charge-storage 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 RCM2300 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 A, B, or C. Only Serial Port B is accessible by using a 10-pin header to DB9 cable with header J6 on the Prototyping Board. 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

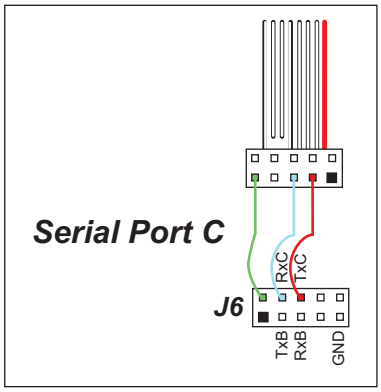
Stop bits: 1

Flow control: None

To access Serial Port A or Serial Port C, change the 2 to 1 or 3 respectively in the following line in the sample program.

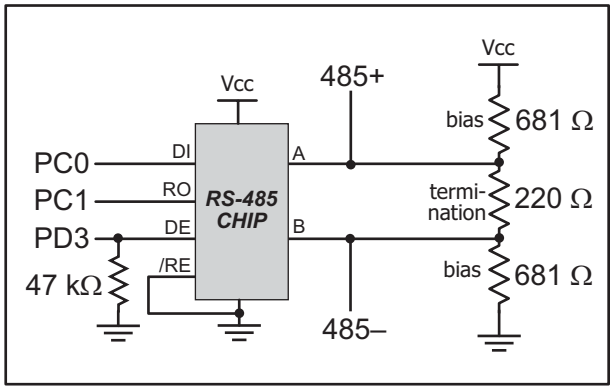
```
#define SERIAL_PORT2
```

You can then access Serial Port A through HyperTerminal or Tera Term using the programming cable's **DIAG** connector to connect the programming header (J1) on the RCM2300 to the PC COM port. Serial Port C can be accessed with your own hookup wire to connect TxC and RxC as shown from header J6 on the Prototyping Board to the 10-pin header to DB9 cable, which is connected to the PC COM port.



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 RCM2300. The slave will send back converted upper case letters back to the master RCM2300 and display them in the **STDIO** window. Use **SLAVE.C** to program the slave RCM2300—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 RCM2300. The slave will send back converted upper case letters back to the master RCM2300 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 Sample Program Descriptions

3.1.3.1 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***.

3.1.3.2 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***.

3.1.3.3 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 RCM2300. Appendix A, "Rabbit-Core RCM2300 Specifications," provides complete physical and electrical specifications.

4.1 RCM2300 Digital Inputs and Outputs

Figure 4 shows the subsystems designed into the RCM2300.

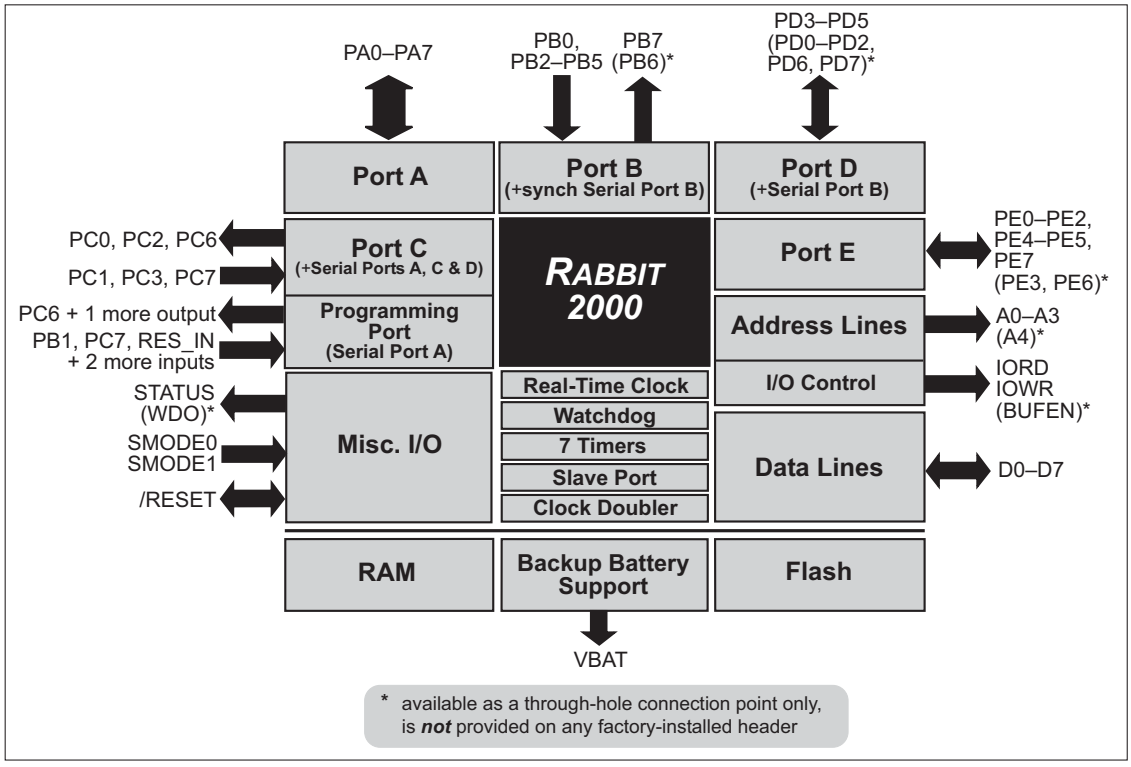


Figure 4. Rabbit Subsystems

The RCM2300 modules have two 26-pin headers to which cables can be connected, or which can be plugged into matching sockets on a production device. The pinouts for these connectors are shown in Figure 5 below.

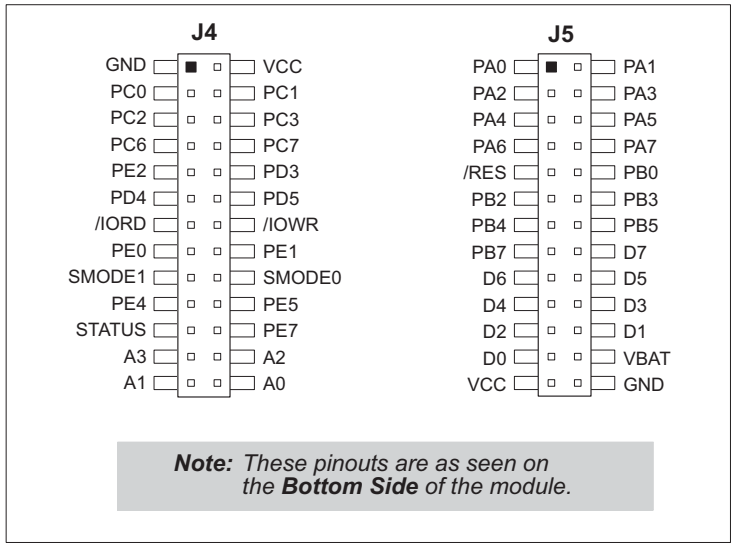


Figure 5. RCM2300 I/O Pinout

Fifteen additional connection points are available along one edge of the RCM2300 board. These connection points are 0.030" diameter holes spaced 0.05" apart. Nineteen additional connection points are available at locations J2 and J3. These additional connection points are reserved for future use.

The remaining discussion is focused on the I/O points available on headers J4 and J5 because it is anticipated that most users will not use the through-hole connection points because of their reduced convenience.

Table 1 lists the pinout configurations on headers J4 and J5. The ports on the Rabbit 2000 microprocessor used in the RCM2300 are configurable, and so the factory defaults can be reconfigured. Table 1 lists the factory defaults and the alternate configurations.

.

Table 1. RabbitCore RCM2300 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	PC6	Output	TXA	Is also connected to programming port used to program/debug
	8	PC7	Input	RXA	
	9	PE2	Bidirectional I/O	I/O control	
	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	SMODE1	Startup mode bit input	Input	Can only be used as general inputs *after* the startup mode op-code has been read following boot-up
	18	SMODE0	Startup mode bit input	Input	
	19	PE4	Bitwise or parallel programmable I/O	I4 control or INT0B input	
	20	PE5		I5 control or INT1B input	
	21	STATUS	Low on first op-code fetch of instruction	Output	Accessed by addressing Global Output Control Register
	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. RabbitCore RCM2300 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			

4.1.1 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 in master/slave communication between two processors. 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, PC3, and PC7 are general CMOS inputs only. These pins can instead be selectively enabled to serve as the serial data inputs for Serial Ports D, C, and A.

SMODE0 and SMODE1 are read at start-up, and set the mode whereby instructions are fetched. Thereafter the user may use and read these pins as inputs by reading the Slave Port Control Register.

> **NOTE:** Exercise care so that the SMODE0 and SMODE1 pins revert to the correct startup code when a reset occurs.

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, PC2, and PC6 are also output-only pins; alternatively, they can serve as the serial data outputs for Serial Ports D, C, and A.

The STATUS pin goes low by default after the first op-code fetch of an instruction cycle. The STATUS pin may be programmed as a separate output by changing the Rabbit 2000's Global Output Control Register.

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.

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.

4.2 Serial Communication

The RCM2300 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 RCM2300 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 sustain their operation in an asynchronous mode up to the baud rate of the system clock divided by 64. The maximum burst rate for an asynchronous byte can be as high as the system clock divided by 32. 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 sustained baud rate can be up to the system clock frequency divided by 8, or 2.76 Mbps for a 22.1 MHz clock speed. The maximum burst rate for a byte can be as high as the system clock divided by 4.

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

4.2.2 Programming Port

The RCM2300 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 RCM2300'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 RCM2300'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 RCM2300's programming header with the RCM2300 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 RCM2300 is automatically in Program Mode when the **PROG** connector on the programming cable is attached to the RCM2300, 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.

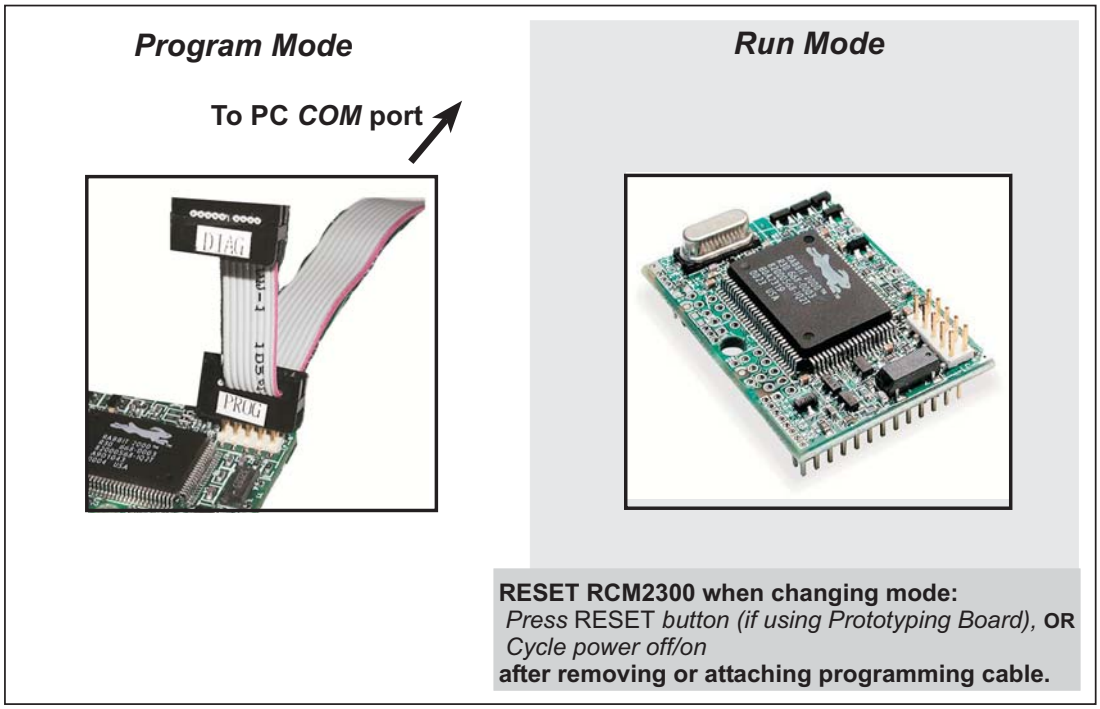


Figure 6. Switching Between Program Mode and Run Mode

A program "runs" in either mode, but can only be downloaded and debugged when the RCM2300 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 RCM2300

The RCM2300 must be programmed via the RCM2200/RCM2300 Prototyping Board or via a similar arrangement on a customer-supplied board. Once the RCM2300 has been programmed successfully, remove the programming cable from the programming connector and reset the RCM2300. The RCM2300 may be reset by cycling the power off/on or by pressing the **RESET** button on the Prototyping Board. The RCM2300 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 RCM2300 module to protect against inadvertent shorts across the pins or damage to the RCM2300 if the pins are not plugged in correctly. Do not reapply power until you have verified that the RCM2300 module is plugged in correctly.

4.4 Other Hardware

4.4.1 Clock Doubler

The RCM2300 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.

4.4.2 Spectrum Spreader

RCM2300 boards with a Rabbit 2000 microprocessor labeled IQ4T or higher have a spectrum spreader, which helps to mitigate EMI problems. By default, the spectrum spreader is on automatically for these boards 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 not necessary for the RCM2300.
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 RCM2300 boards with Rabbit 2000 chips labeled *IQ3T* or earlier, or with a version of Dynamic C prior to 7.30.

4.5 Memory

4.5.1 SRAM

The RCM2300 is designed to accept 128K of SRAM packaged in an SOIC case.

4.5.2 Flash EPROM

The RCM2300 is also designed to accept 128K to 512K of flash EPROM packaged in a TSOP case.

> **NOTE:** Rabbit Semiconductor 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. 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.

4.5.3 Dynamic C BIOS Source Files

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

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 Semiconductor single-board computers and other single-board computers based on the Rabbit microprocessor. Chapter 4 provides the libraries, function calls, and sample programs related to the RCM2300.

5.1 More About Dynamic C

Dynamic C has been in use worldwide since 1989. It is specially designed for programming embedded systems, and features quick compile and interactive debugging in the real environment. A complete reference guide 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 RCM2300. The advantage of working in RAM is to save wear on the flash memory, which is limited to about 100,000 write cycles.

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

NOTE: Do not depend on the flash memory sector size or type. Due to the volatility of the flash memory market, the RCM2300 and Dynamic C were designed to accommodate flash devices with various sector sizes.

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, 98, 2000, NT, Me, and XP. 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 RCM2300 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/RCM2300** directory provide further examples.

These functions are provided for convenience, not speed. User code should be written in assembly language when speed is important.

5.2.1 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

The Prototyping Board has room for an RS-232 chip. Dynamic C has two libraries to support serial communication: **RS232.LIB** provides a set of circular-buffer-based functions, and **PACKET.LIB** provides packet-based support. Packets can be delimited by time gap, 9th bit detection, or special-character detection.

Both the packet-based and the circular-buffer-based routines are available in blocking and nonblocking (cofunction) flavors. See the ***Dynamic C User's Manual*** and Technical Note 213, ***Rabbit 2000 Serial Port Software***, for more details on serial communication.

5.4 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 Semiconductor 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.

5.4.1 Upgrades

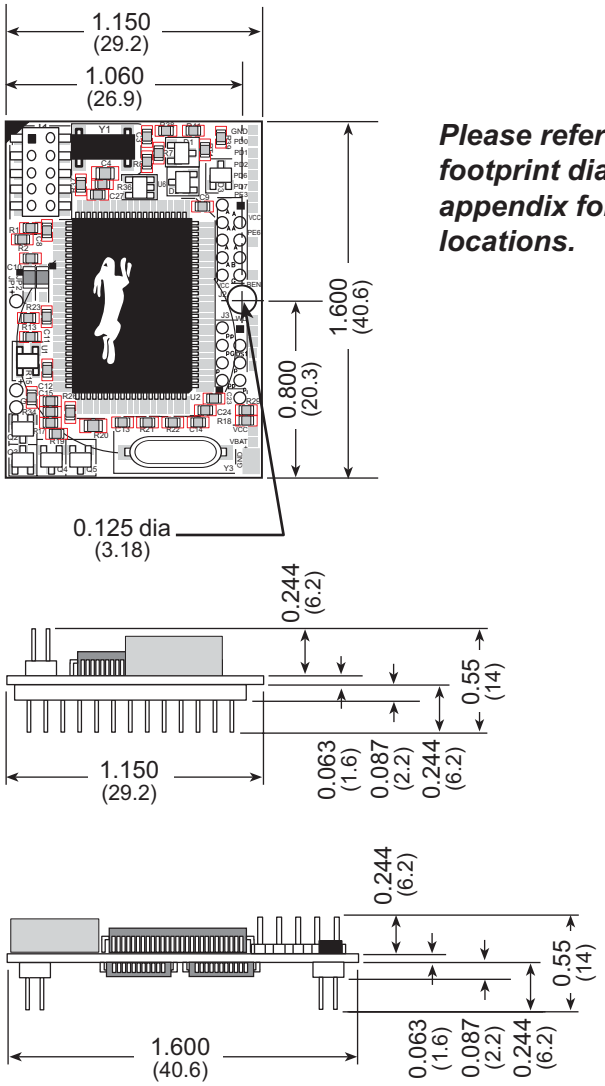
Dynamic C installations are designed for use with the board they are included with, and are included at no charge as part of our low-cost kits. Dynamic C is a complete software development system, but does not include all the Dynamic C features. Rabbit Semiconductor also offers add-on Dynamic C modules containing the popular µC/OS-II real-time operating system, as well as PPP, Advanced Encryption Standard (AES), and other select libraries. In addition to the Web-based technical support included at no extra charge, a one-year telephone-based technical support module is also available for purchase.

APPENDIX A. RABBITCORE RCM2300 SPECIFICATIONS

Appendix A provides the specifications for the RCM2300, and describes the conformal coating.

A.1 Electrical and Mechanical Characteristics

Figure A-1 shows the mechanical dimensions for the RCM2300.



Please refer to the RCM2300 footprint diagram later in this appendix for precise header locations.

Figure A-1. RabbitCore RCM2300 Dimensions

NOTE: All measurements are in inches followed by millimeters enclosed in parentheses. All dimensions have a manufacturing tolerance of ±0.01" (0.2 mm).

It is recommended that you allow for an “exclusion zone” of 0.04" (1 mm) around the RCM2300 in all directions when the RCM2300 is incorporated into an assembly that includes other printed circuit boards. An “exclusion zone” of 0.12" (3 mm) is recommended below the RCM2300 when the RCM2300 is plugged into another assembly using the shortest connectors for headers J4 and J5. Figure A-2 shows this “exclusion zone.”

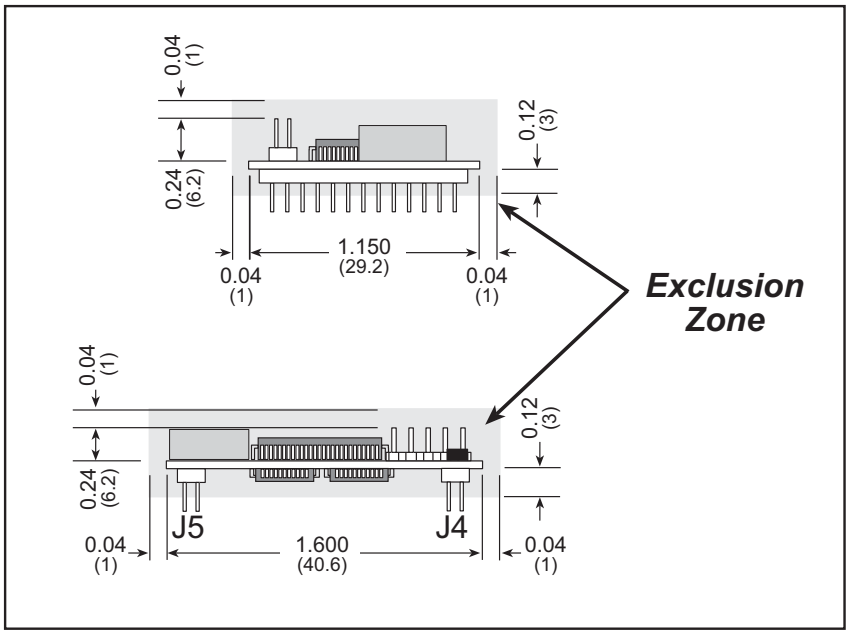


Figure A-2. RCM2300 “Exclusion Zone”

Table A-1 lists the electrical, mechanical, and environmental specifications for the RCM2300.

Table A-1. RabbitCore RCM2300 Specifications

Parameter	Specification
Microprocessor	Rabbit 2000® at 22.1 MHz
Flash Memory	256K
SRAM	128K
Backup Battery	Connection for user-supplied backup battery (to support RTC and SRAM)
General-Purpose I/O*	29 parallel I/0 lines grouped in five 8-bit ports (shared with serial ports): • 17 configurable I/O • 8 fixed inputs • 4 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 (extra address and buffer enable via separate connections)
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	Max. burst rate = CLK/32 Max. sustained rate = CLK/64
Slave Interface	A slave port allows the RCM2300 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, 108 mA
Operating Temperature	–40°C to +85°C
Humidity	5% to 95%, noncondensing
Connectors	Two IDC headers 2 × 13, 2 mm pitch
Board Size	1.15" × 1.60" × 0.55" (29 mm × 41 mm × 14 mm)

* 15 additional I/O are available via less convenient 0.03" diameter through-hole connection points

A.1.1 Headers

The RCM2300 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 footprint of another board that the RCM2300 would be plugged into. These values are relative to the header connectors.

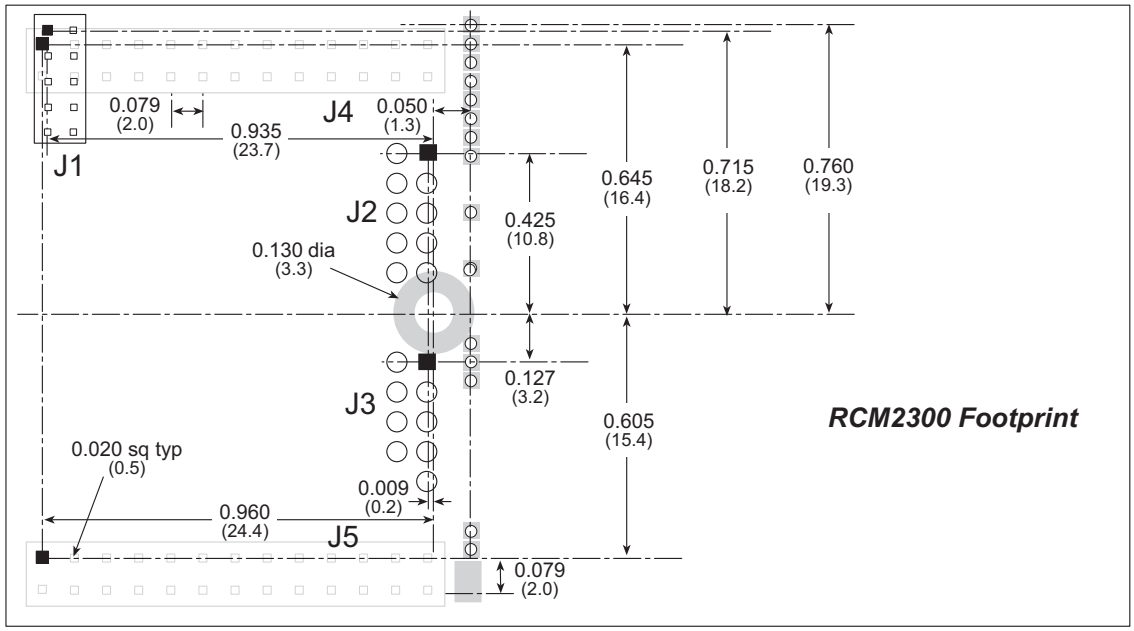


Figure A-3. User Board Footprint for RabbitCore RCM2300

A.1.2 Physical Mounting

An 9/32" or ¼" (7 mm) metal standoff with insulating washers and a 4-40 screw is recommended to attach the RCM2300 to a user board at the hole position shown in Figure A-3.

A.2 Bus Loading

You must pay careful attention to bus loading when designing an interface to the RCM2300. This section provides bus loading information for external devices.

Table A-2 lists the capacitance for the various Rabbit 2000 I/O ports with SRAM and flash memory connected.

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

I/O Ports	Input Capacitance (pF)	Output Capacitance (pF)
Parallel Ports A to E	12	14
Data Lines D0–D7	30	32
Address Lines A0–A12	—	32

Table A-3 lists the external capacitive bus loading for the various Rabbit 2000 output ports. Be sure to add the loads for the devices you are using in your system and verify that they do not exceed the values in Table A-3.

Table A-3. External Capacitive Bus Loading -40°C to +85°C

Output Port	Clock Speed (MHz)	Maximum External Capacitive Loading (pF)
A[4:1] D[7:1]	22.1	50
A[4:1] D[7:1]	22.1	100 for 55 ns flash
A0 D0	22.1	100
PD[3:0]	22.1	100
PA[7:0] PB[7,6] PC[6,2,0] PD[7:0] PE[7:0]	22.1	90
All data, address, and I/O lines with clock doubler disabled	11.06	100

The values from the table above are derived using 55 ns (flash memory) and 70 ns (SRAM) memory access times. External capacitive loading can be improved by 10 pF for commercial temperature ranges, but do not exceed 100 pF. See the AC timing specifications in the ***Rabbit 2000 Microprocessor Users Manual*** for more information.

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

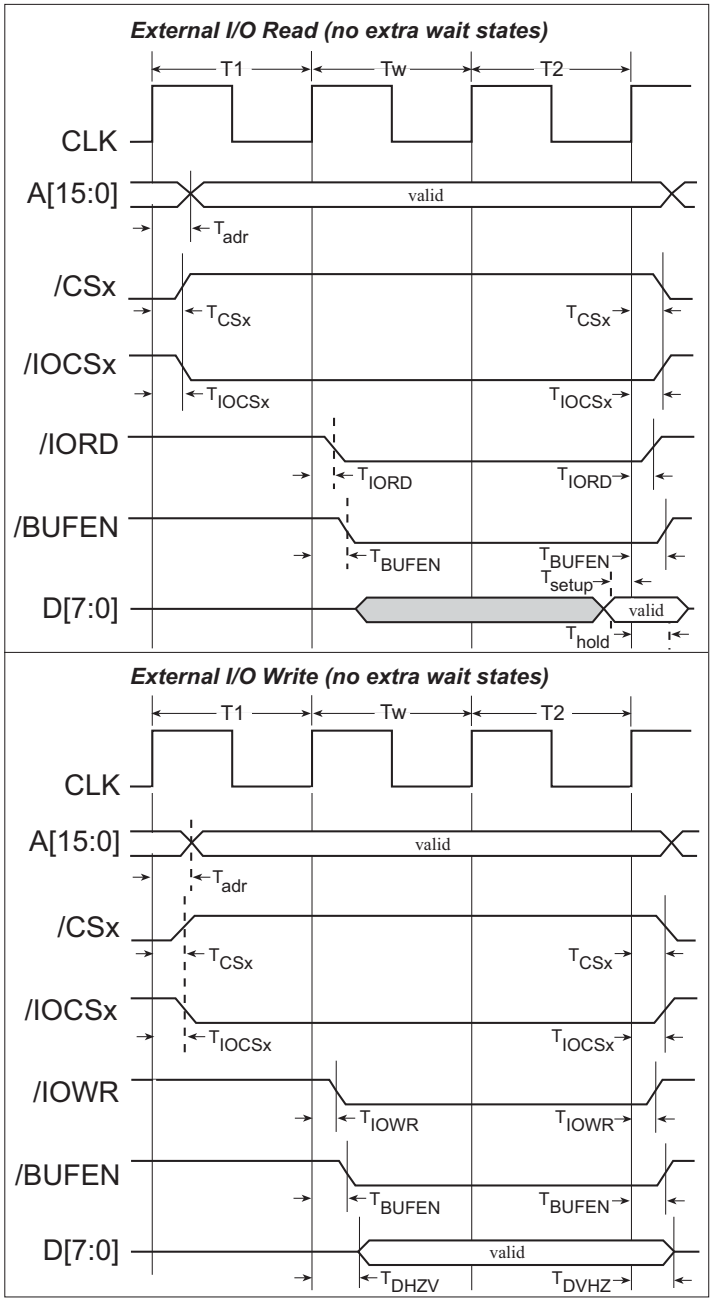


Figure A-4. External I/O Read and Write Cycles—No Extra Wait States

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. T_{setup} is specified from 30%/70% of the V_{DD} voltage level.

Table A-4 lists the parameters shown in these figures and provides minimum or measured values.

Table A-4. 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 (10 ns @ 3.3 V) 14 ns @ 70 pF, 5 V (19 ns @ 3.3 V)
	T_{setup}	Data read setup time	Min.	2 ns @ 5 V (3 ns @ 3.3 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 (10 ns @ 3.3 V) 14 ns @ 70 pF, 5 V (19 ns @ 3.3 V)
	T_{hold}	Data write hold time from /WEx or /IOWR	Min.	½ CPU clock cycle

A.3 Rabbit 2000 DC Characteristics

Table A-5 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-5. 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-6 (sinking) V_{DD} = 4.5 V		0.2	0.4	V
V_{OH}	CMOS Output High Voltage	I_{OH} = See Table A-6 (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 22.1 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–A4 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-6 shows the AC and DC output drive limits of the parallel I/O buffers when the Rabbit 2000 is used in the RCM2300.

Table A-6. 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, 1, 0]	8/8	12/12
PC [6, 2, 0]	8/8	12/12
PD [7:4]	8/8	12/12
PD [3:0]**	16/16	25/25
PE [7:0]	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 the first footnote.

A.5 Conformal Coating

The area around the crystal oscillator has had the Dow Corning silicone-based 1-2620 conformal coating applied. The conformally coated area is shown in Figure A-5. The conformal coating protects these high-impedance circuits from the effects of moisture and contaminants over time.

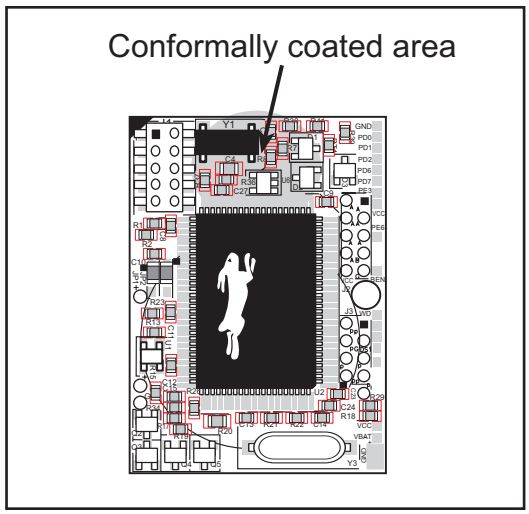


Figure A-5. RCM2300 Areas Receiving Conformal Coating

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

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

A.6 Jumper Configurations

Figure A-6 shows the header locations used to configure the various RCM2300 options via jumpers.

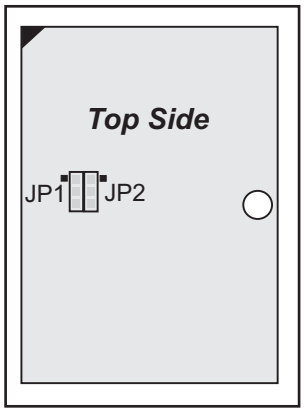


Figure A-6. Location of RCM2300 Configurable Positions

Table A-7 lists the configuration options.

Table A-7. RCM2300 Jumper Configurations

Header	Description	Pins Connected		Factory Default
JP1	Flash Memory Size	1–2	128K/256K	×
		2–3	512K	
JP2	Flash Memory Bank Select	1–2	Normal Mode	×
		2–3	Bank Mode	

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

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 RCM2300 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 RCM2300 to a power supply for development. It also provides some basic I/O peripherals (switches and LEDs), as well as a prototyping area for more advanced hardware development.

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

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

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

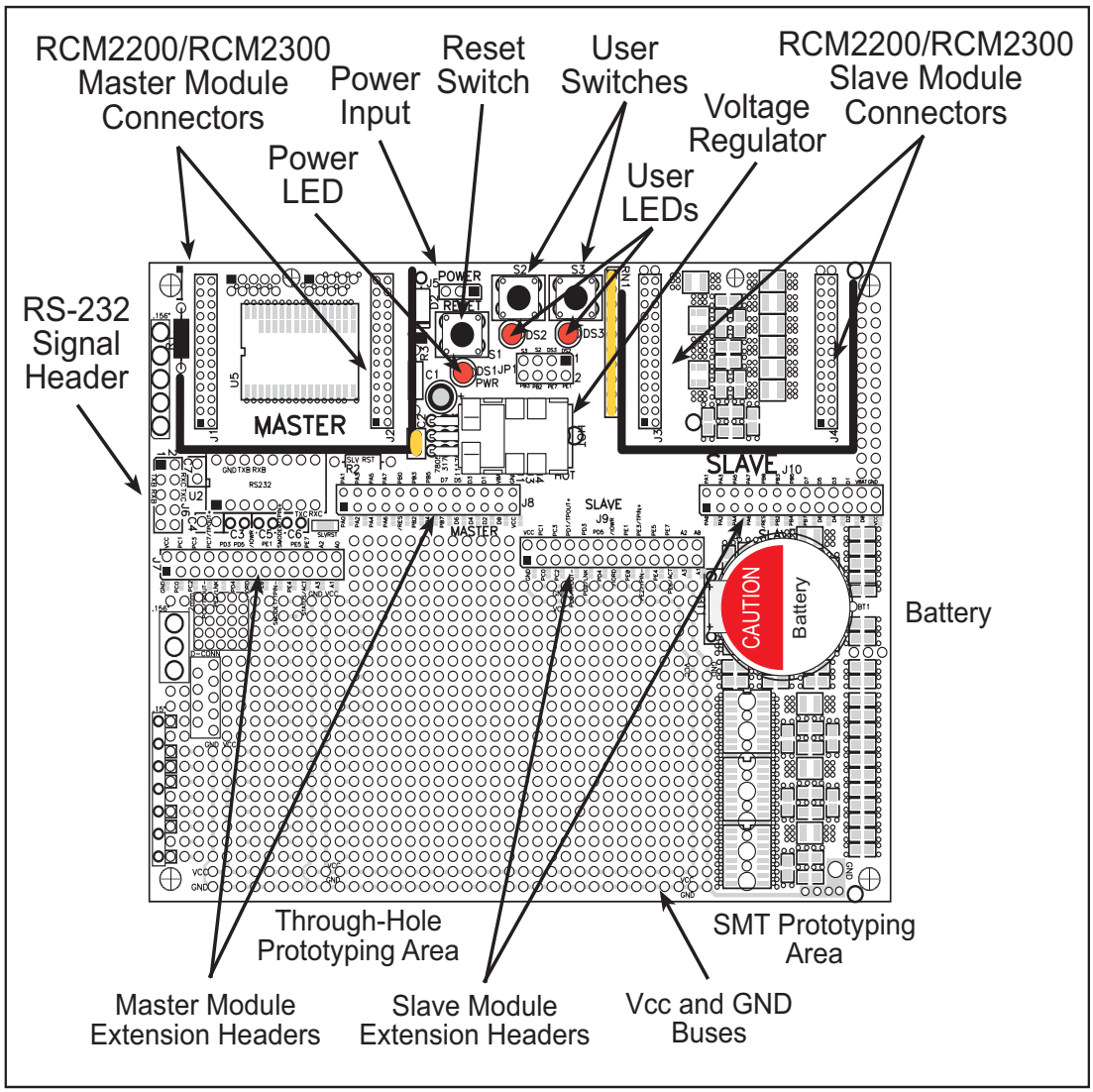


Figure B-1. RCM2200/RCM2300 Prototyping Board

B.1.1 Prototyping Board Features

- **Power Connection**—A 3-pin header is provided at J5 for the power supply connection. Note that both outer pins are connected to ground and the center pin is connected to the raw V+ input. The cable from the wall transformer provided with the North American version of the Development Kit ends in a connector that may be connected in either orientation.

  Users providing their own power supply should ensure that it delivers 7.5–25 V DC at not less than 500 mA. The voltage regulator will get warm in use. (Lower supply voltages will reduce thermal dissipation from the device.)

- **Regulated Power Supply**—The raw DC voltage provided to the **POWER** header at J5 is routed to a 5 V linear voltage regulator, which provides stable power to the RCM2300 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 master RCM2300'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 RCM2300, and may be read as inputs by sample applications.

  Two LEDs are connected to the PE1 and PE7 pins of the master RCM2300, 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 may then be 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, which must be soldered once it is in the hole.

- **Master Module Connectors**—When the RCM2300 plugged into the **MASTER** slots, it can act as the "master" relative to another RabbitCore RCM2200 or RCM2300 plugged into the **SLAVE** slots, which acts as the "slave."

  This master/slave relationship is ***not*** used in the DeviceMate Development Kit where the "target" RCM2300 is plugged into the **MASTER** slots, and the RCM2200, which is used as the DeviceMate hardware platform, is plugged into the **SLAVE** slots. The Prototyping and Demonstration Board serves only as a means to connect the two RabbitCore modules together to demonstrate the DeviceMate software features in Dynamic C.

- **Slave Module Connectors**—A second set of connectors is pre-wired to permit installation of a second, slave RCM2200 or RCM2300.

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 in the ***RCM2300 Getting Started Manual***, you may wish to expand the Prototyping 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 set of both the master and slave modules is duplicated at these two sets of headers. Developers can solder wires directly into the appropriate holes, or, for more flexible development, 0.1" pitch 26-pin header strips can be soldered into place. See Figure B-5 for the header pinouts.
- **RS-232**—Two 2-wire or one 5-wire RS-232 serial port can be added to the Prototyping Board by installing an RS-232 driver IC and four capacitors. The Maxim MAX232CPE driver chip or a similar device is recommended for 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 top side of the Prototyping Board below and to the left of the **MASTER** module position.

  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 RCM2300 module are hard-wired to the Prototyping Board LEDs and switches through JP1 on the underside of the Prototyping Board.

B.2 Mechanical Dimensions and Layout

Figure B-2 shows the mechanical dimensions and layout for the RCM2300 Prototyping Board.

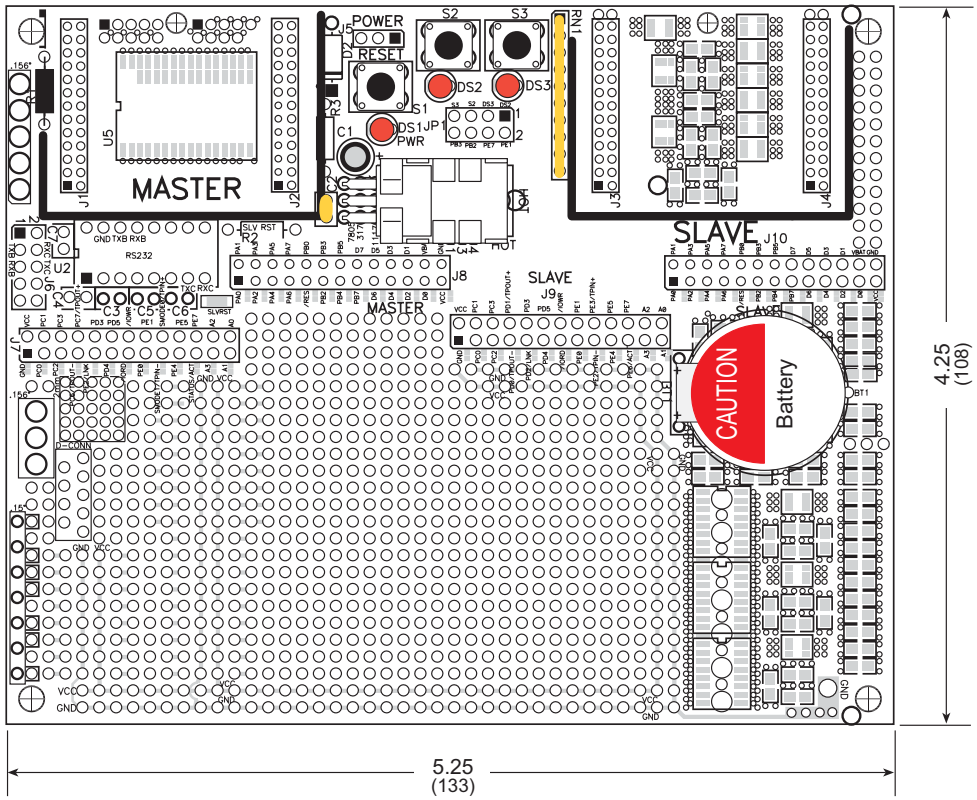


Figure B-2. RCM2300 Prototyping Board Dimensions

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

Table B-1. RCM2300 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
Corner Standoffs/Spacers	4, accept 6-32 × 3/8 screws

B.3 Power Supply

The RCM2300 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.

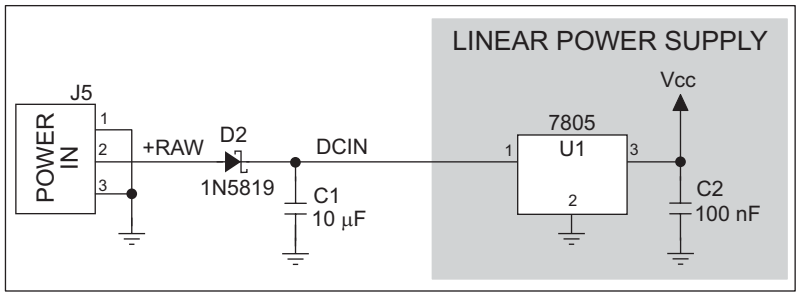


Figure B-3. Prototyping Board Power Supply

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 RCM2300 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 RCM2300. 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 RCM2300.

To maximize the availability of RCM2300 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. 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.

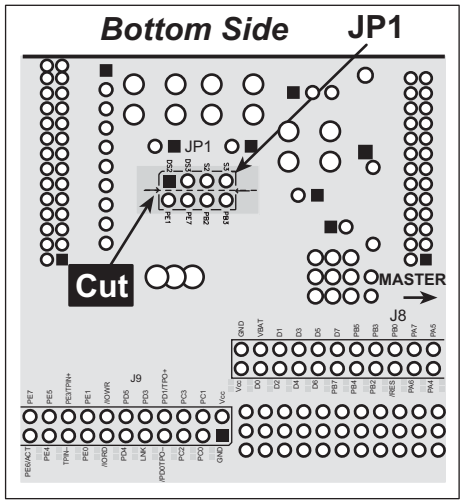


Figure B-4. Where to Cut Traces to Permanently Disable Demonstration Hardware on Prototyping Board

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 RCM2300 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.

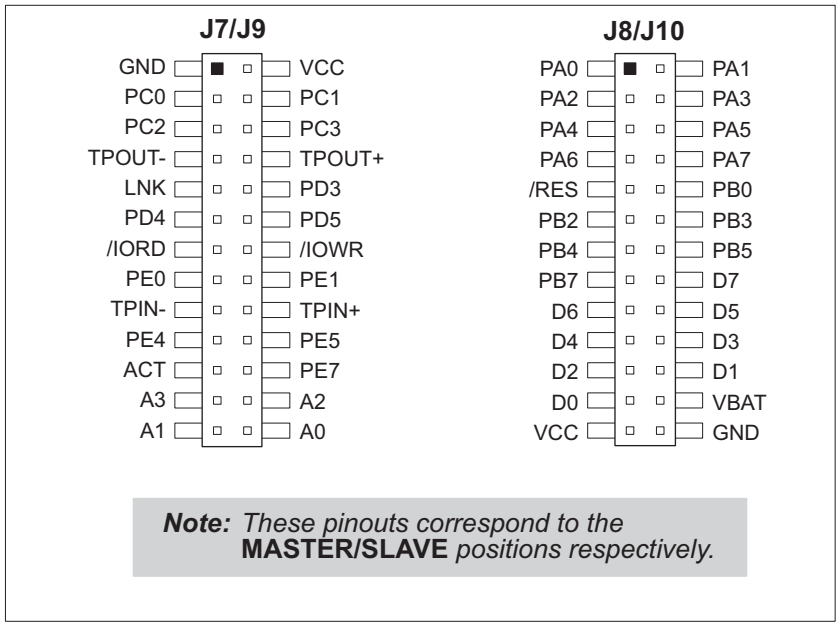


Figure B-5. RCM2300 Prototyping Board Pinout (Top View)

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.

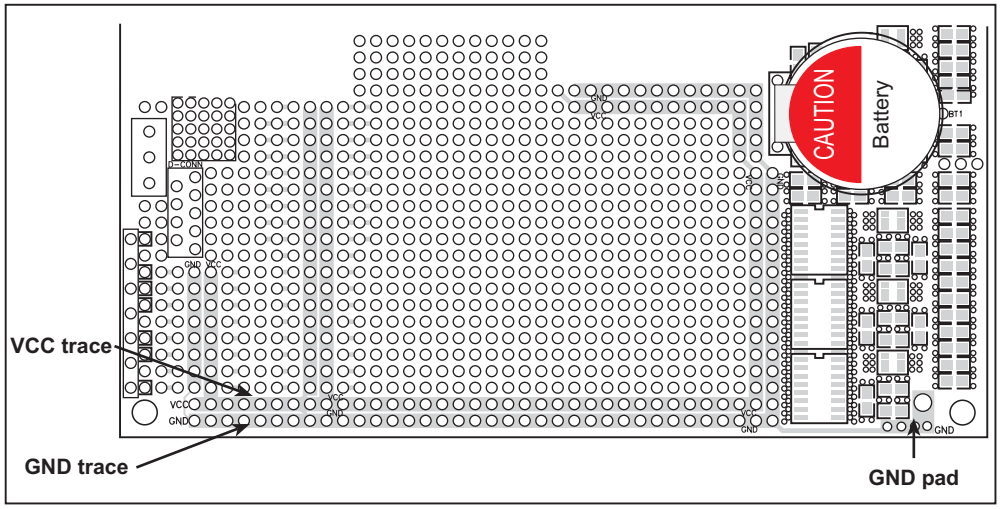


Figure B-6. VCC and GND Traces Along Edge of Prototyping Board

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.

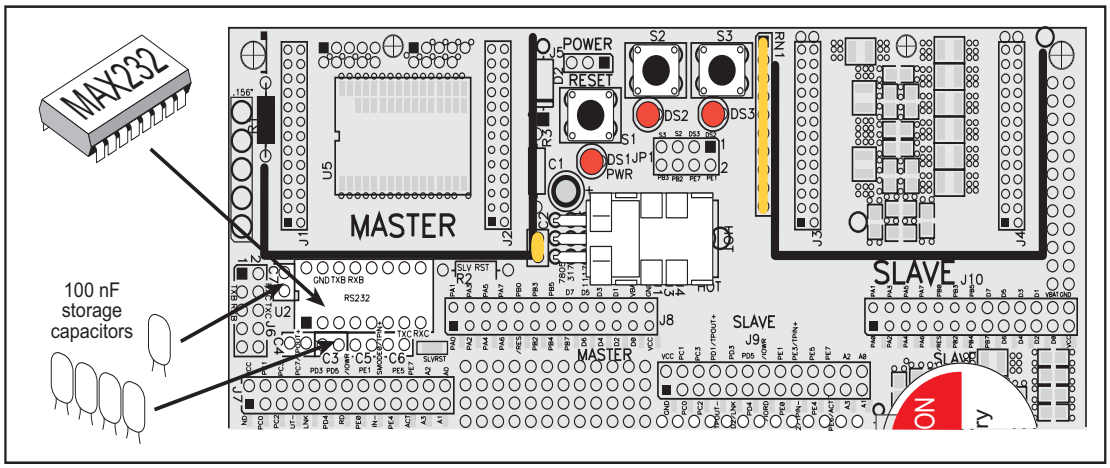


Figure B-7. Location for User-Supplied RS-232 Transceiver and Charge Storage Capacitors on Back Side of Prototyping Board

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 at the lower right corner of the Prototyping Board 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 RCM2300, and some background on the chip select circuit used in power management.

C.1 Power Supplies

The RCM2300 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 RCM2300 with no loading at the outputs operating at 22.1 MHz typically draws 108 mA. The RCM2300 will consume an additional 10 mA when the programming cable is used to connect J1 to a PC.

C.2 Battery Backup

The RCM2300 does not have a factory-installed 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 an external battery. This header makes it possible to connect an external 3 V power supply.

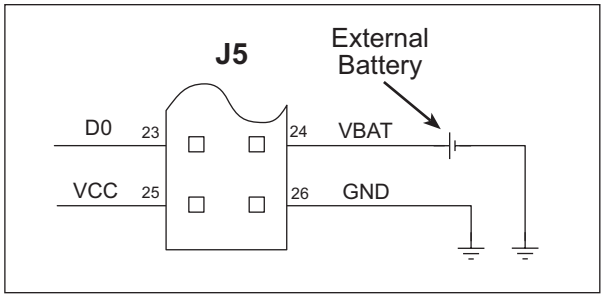


Figure C-1. External Battery Connections at Header J5

The RCM2300 has another battery option available. A customer-installed BR2577A/GA backup battery can be soldered right on the RCM2300 as shown in Figure C-2. The negative battery connection is to the pin 3 hole in the area corresponding to header area J3.

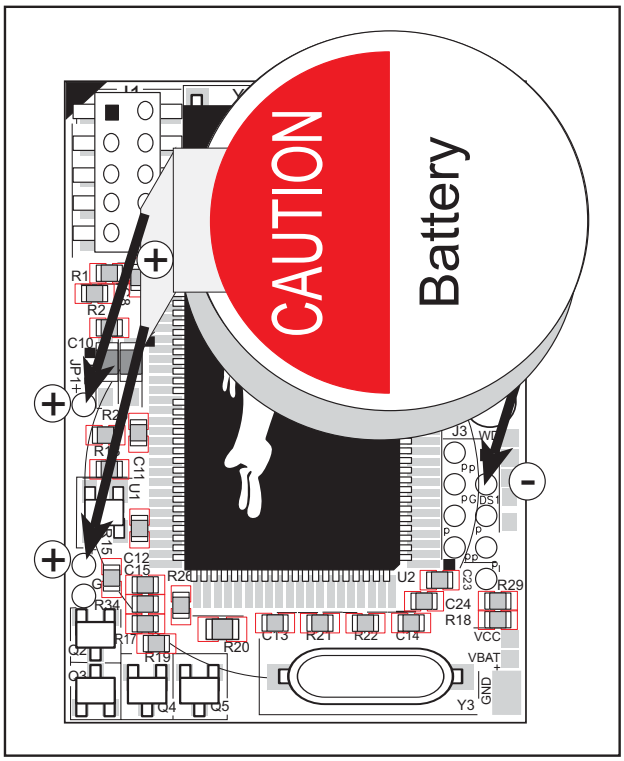


Figure C-2. Installing Onboard Backup Battery on RCM2300

NOTE: Installing an onboard backup battery directly on the RCM2300 will prevent you from adding a through-hole connector at position J3 pin 3 on the other side of the RCM2300.

Alternatively, you may wish to add a 2-pin connector with a 2 mm pitch for hooking up to an external backup battery as shown in Figure C-3.

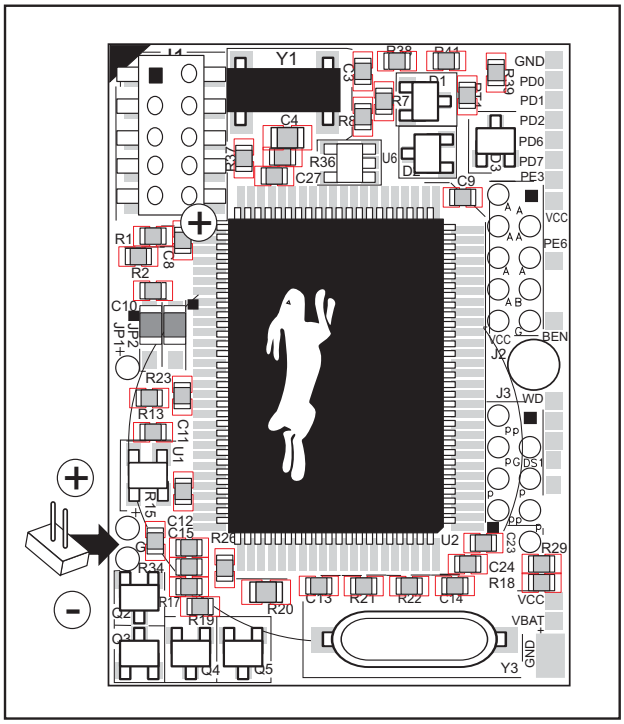


Figure C-3. Installing Optional Battery Connector on RCM2300

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 RCM2300 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 RCM2300 and the storage capacity of the battery. Note that the shelf life of a lithium battery is ultimately 10 years.

C.2.1 Battery Backup Circuits

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 RCM2300.

Figure C-4 shows the RCM2300 battery-backup circuit.

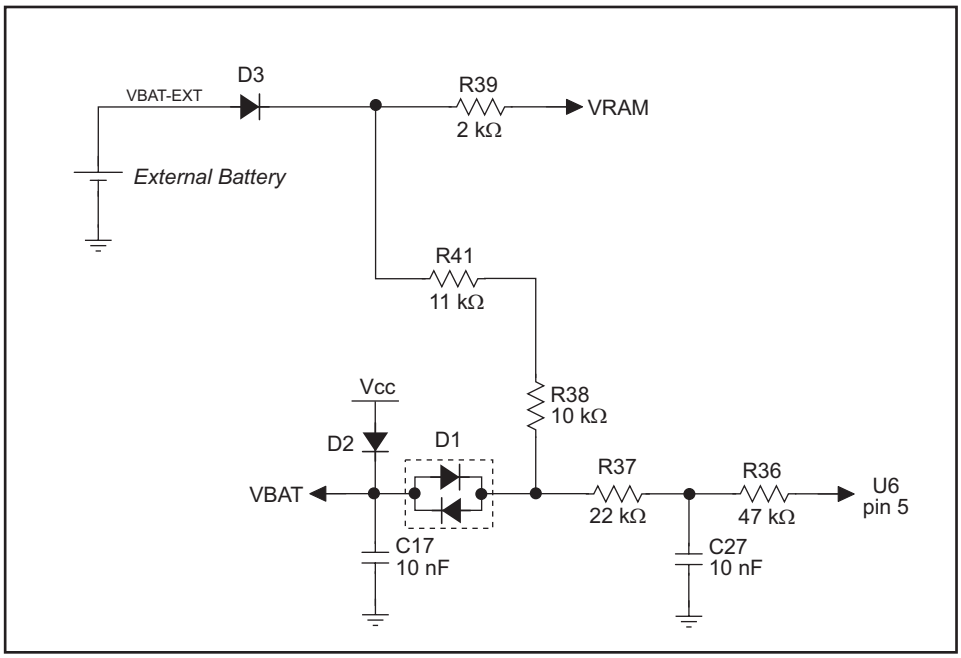


Figure C-4. RCM2300 Battery-Backup Circuit

C.2.2 Reset Generator

The RCM2300 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 RCM2300 has a reset output, pin 9 on header J5. This reset output can be sensed externally. The output can also be overridden and forced into any state by using a circuit capable of providing 5 mA of output current.

C.3 Chip Select Circuit

The RCM2300 has provision for battery backup, which kicks in to keep VRAM from dropping below 2 V.

When the RCM2300 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-5 shows a schematic of the chip select control switch.

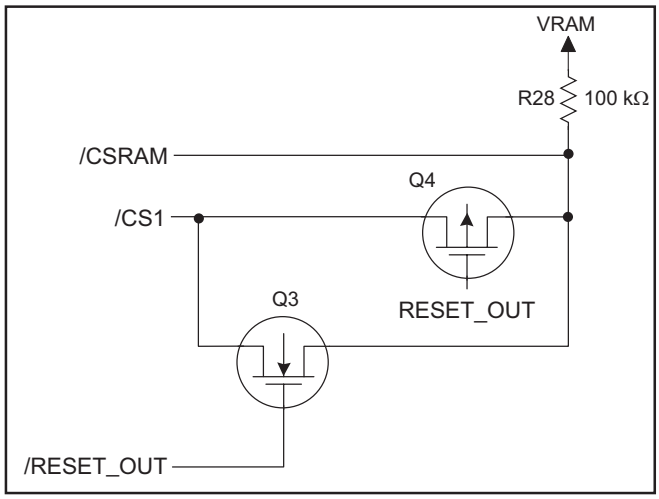


Figure C-5. Chip Select Control Switch

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 voltage 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 RCM2300.

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

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

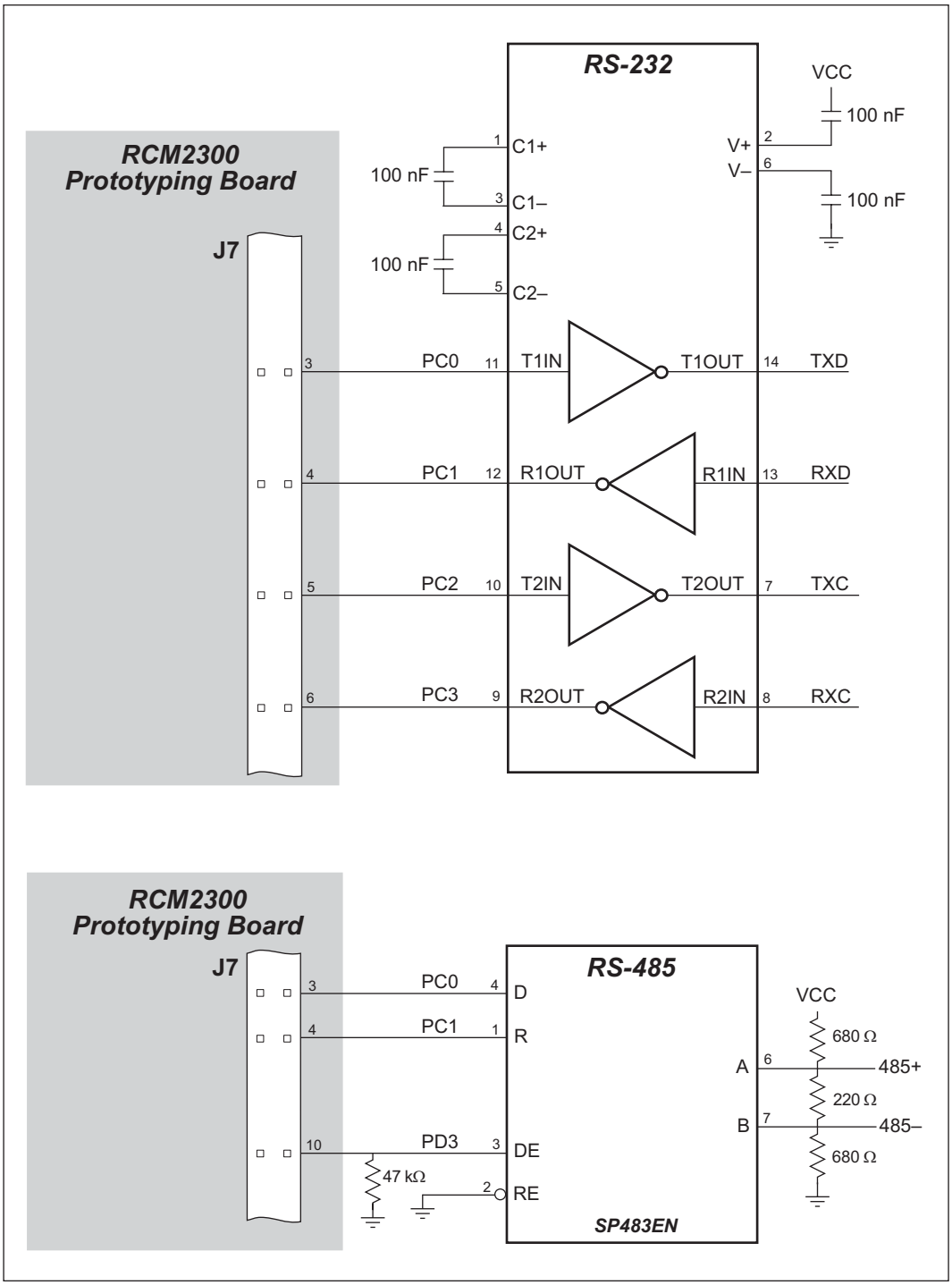


Figure D-1. Sample RS-232 and RS-485 Circuits

Sample Program: **PUTS.C** in **SAMPLES/RCM2300**.

D.2 Keypad and LCD Connections

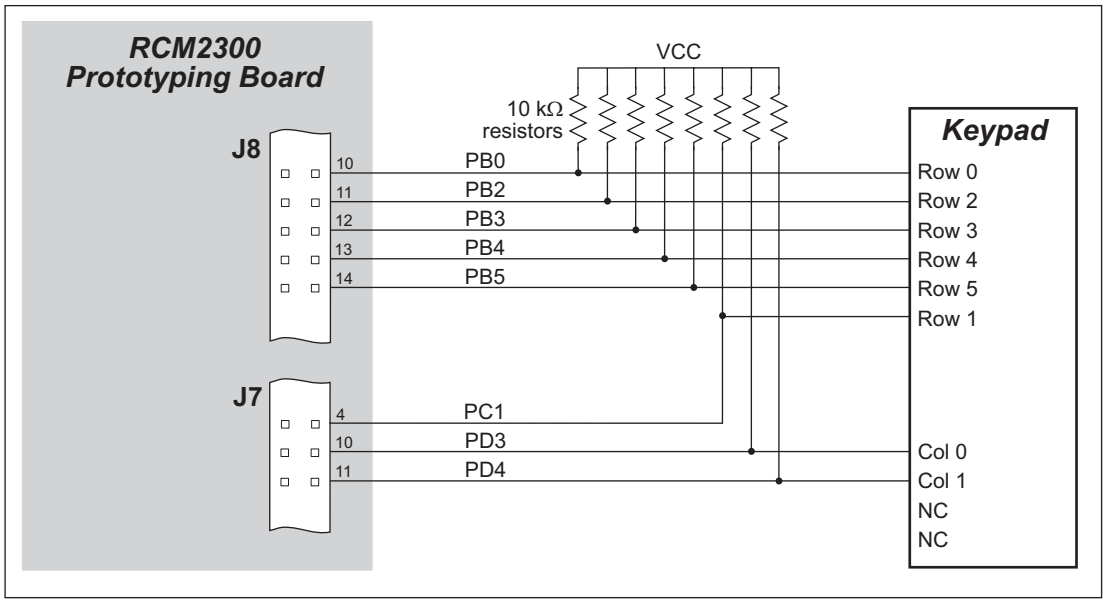


Figure D-2. Sample Keypad Connections

Sample Program: **KEYLCD.C** in **SAMPLES/RCM2300**.

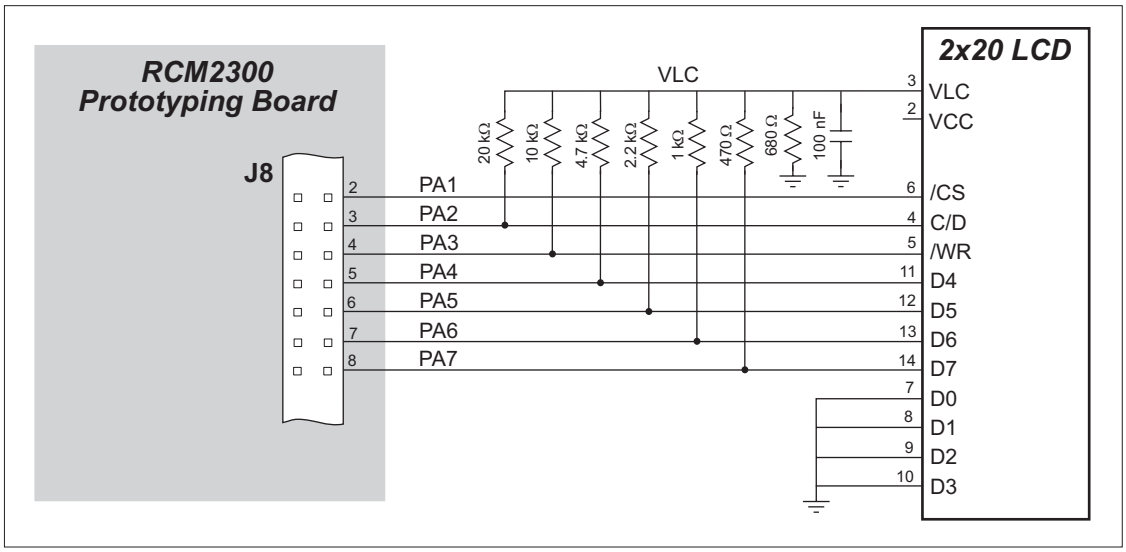


Figure D-3. Sample LCD Connections

Sample Program: **KEYLCD.C** in **SAMPLES/RCM2300**. (When Parallel Port A is not being used for quick communication, its resting, quiescent value is used to set the LCD contrast level.)

D.3 External Memory

The sample circuit can be used to access 16 bytes on 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 to create up to four thousand 16-byte pages.

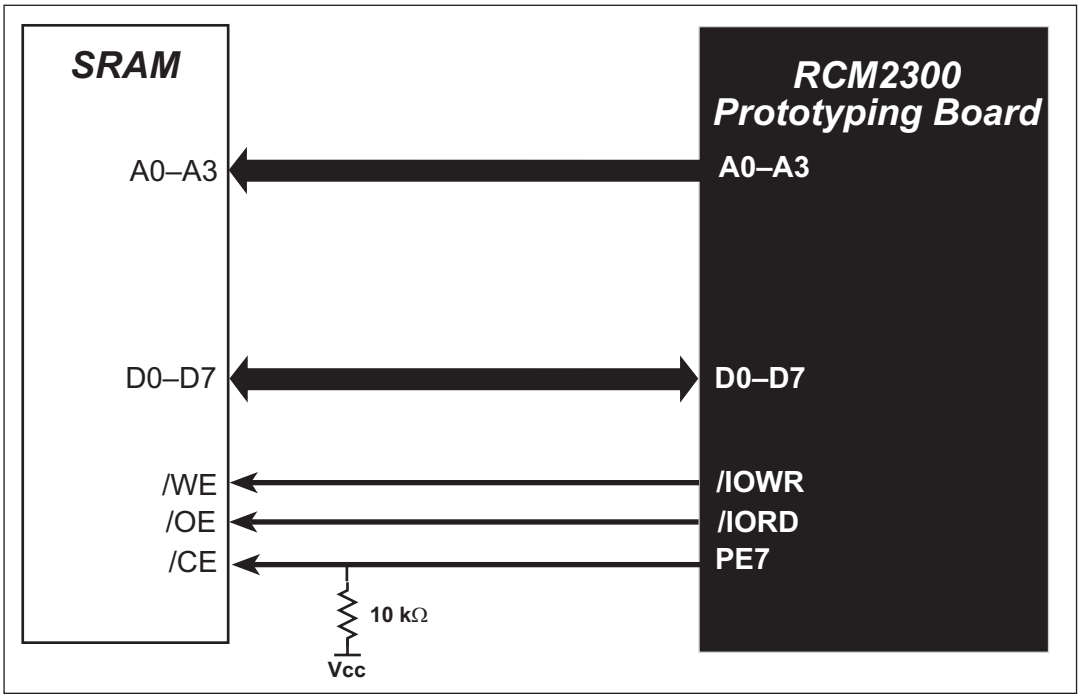


Figure D-4. Sample External Memory Connections

Sample Program: **EXTSRAM.C** in **SAMPLES/RCM2300**.

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.

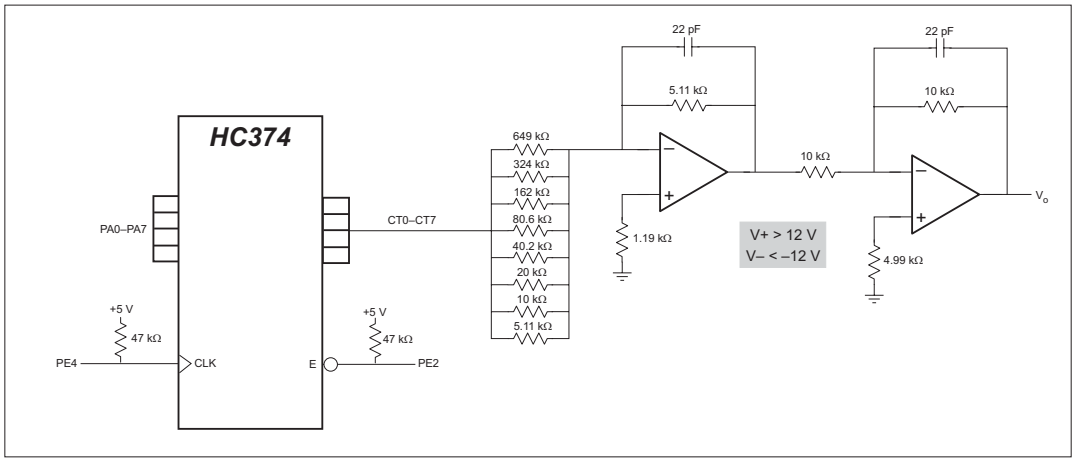


Figure D-5. Sample D/A Converter Connections

INDEX

R

S

T

U

SCHEMATICS

090-0119 RCM2300 Schematic

www.rabbit.com/documentation/schemat/090-0119.pdf

090-0122 RCM2200/RCM2300 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.