



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



## Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China





# **RabbitCore RCM4300/ RCM4310/RCM4320**

C-Programmable Analog Core Module  
with microSD Card Storage and Ethernet

## **User's Manual**

019-0163\_K

RabbitCore RCM4300/RCM4310/RCM4320 User's Manual

Part Number 019-0163\_K • Printed in U.S.A.

©2007–2017 Digi International Inc. • All rights reserved.

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

#### Trademarks

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

Rabbit 4000 is a trademark of Digi International Inc.

SD is a trademark of the SD Card Association.

The latest version of this document is available at [www.digi.com/support](http://www.digi.com/support).

Digi International Inc.

[www.digi.com](http://www.digi.com)

---

# TABLE OF CONTENTS

<b>Chapter 1. Introduction</b>	<b>6</b>
1.1 RCM4300 Features .....	7
1.2 Advantages of the RCM4300 .....	8
1.3 Development and Evaluation Tools.....	9
1.3.1 RCM4300 Development Kit .....	9
1.3.2 Software .....	10
1.3.3 Online Documentation .....	10
<b>Chapter 2. Getting Started</b>	<b>11</b>
2.1 Install Dynamic C .....	11
2.2 Hardware Connections.....	12
2.2.1 Step 1 — Prepare the Prototyping Board for Development.....	12
2.2.2 Step 2 — Attach Module to Prototyping Board.....	13
2.2.3 Step 3 — Connect Programming Cable.....	14
2.2.4 Step 4 — Connect Power .....	15
2.3 Run a Sample Program .....	16
2.3.1 Troubleshooting .....	16
2.4 Where Do I Go From Here? .....	17
2.4.1 Technical Support .....	17
<b>Chapter 3. Running Sample Programs</b>	<b>18</b>
3.1 Introduction.....	18
3.2 Sample Programs .....	19
3.2.1 Tamper Detection.....	20
3.2.2 Use of microSD Cards .....	20
3.2.3 Serial Communication.....	21
3.2.4 A/D Converter Inputs (RCM4300 only).....	24
3.2.4.1 Downloading and Uploading Calibration Constants.....	25
3.2.5 Real-Time Clock .....	27
3.2.6 TCP/IP Sample Programs .....	27
3.2.7 FAT File System Sample Programs.....	27
<b>Chapter 4. Hardware Reference</b>	<b>28</b>
4.1 RCM4300 Digital Inputs and Outputs .....	29
4.1.1 Memory I/O Interface .....	35
4.1.2 Other Inputs and Outputs .....	35
4.2 Serial Communication .....	36
4.2.1 Serial Ports .....	36
4.2.2 Ethernet Port .....	37
4.2.3 Programming Port .....	38
4.3 Programming Cable .....	39
4.3.1 Changing Between Program Mode and Run Mode .....	39
4.3.2 Standalone Operation of the RCM4300.....	40

4.4 A/D Converter (RCM4300 only).....	41
4.4.1 A/D Converter Power Supply .....	43
4.5 Other Hardware.....	44
4.5.1 Clock Doubler .....	44
4.5.2 Spectrum Spreader .....	44
4.6 Memory.....	45
4.6.1 SRAM .....	45
4.6.2 Flash Memory .....	45
4.6.3 VBAT RAM Memory .....	45
4.6.4 microSD Cards.....	45
<b>Chapter 5. Software Reference</b> .....	<b>47</b>
5.1 More About Dynamic C .....	47
5.2 Dynamic C Function Calls.....	49
5.2.1 Digital I/O .....	49
5.2.2 Serial Communication Drivers.....	49
5.2.3 Serial Flash Memory Use.....	50
5.2.4 User Block.....	52
5.2.5 SRAM Use .....	52
5.2.6 RCM4300 Cloning.....	53
5.2.7 microSD Card Drivers .....	53
5.2.8 Prototyping Board Function Calls.....	54
5.2.8.1 Board Initialization.....	54
5.2.8.2 Alerts .....	55
5.2.9 Analog Inputs (RCM4300 only).....	56
5.3 Upgrading Dynamic C .....	73
5.3.1 Add-On Modules.....	73
<b>Chapter 6. Using the TCP/IP Features</b> .....	<b>74</b>
6.1 TCP/IP Connections .....	74
6.2 TCP/IP Primer on IP Addresses.....	76
6.2.1 IP Addresses Explained .....	78
6.2.2 How IP Addresses are Used.....	79
6.2.3 Dynamically Assigned Internet Addresses .....	80
6.3 Placing Your Device on the Network.....	81
6.4 Running TCP/IP Sample Programs .....	82
6.4.1 How to Set IP Addresses in the Sample Programs .....	83
6.4.2 How to Set Up your Computer for Direct Connect .....	84
6.5 Run the PINGME.C Sample Program .....	85
6.6 Running Additional Sample Programs With Direct Connect.....	85
6.7 Where Do I Go From Here? .....	86
<b>Appendix A. RCM4300 Specifications</b> .....	<b>87</b>
A.1 Electrical and Mechanical Characteristics .....	88
A.1.1 A/D Converter.....	92
A.1.2 Headers.....	93
A.2 Rabbit 4000 DC Characteristics.....	94
A.3 I/O Buffer Sourcing and Sinking Limit .....	95
A.4 Bus Loading.....	95
A.5 Jumper Configurations.....	98
A.6 Conformal Coating.....	100
<b>Appendix B. Prototyping Board</b> .....	<b>101</b>
B.1 Introduction.....	102
B.1.1 Prototyping Board Features.....	103
B.2 Mechanical Dimensions and Layout.....	105
B.3 Power Supply .....	106

B.4 Using the Prototyping Board.....	107
B.4.1 Adding Other Components.....	109
B.4.2 Measuring Current Draw.....	109
B.4.3 Analog Features (RCM4300 only).....	110
B.4.3.1 A/D Converter Inputs.....	110
B.4.3.2 Thermistor Input.....	112
B.4.3.3 A/D Converter Calibration.....	112
B.4.4 Serial Communication.....	113
B.4.4.1 RS-232.....	114
B.5 Prototyping Board Jumper Configurations.....	115
<b>Appendix C. Power Supply</b> .....	<b>118</b>
C.1 Power Supplies.....	118
C.1.1 Battery Backup.....	118
C.1.2 Battery-Backup Circuit.....	119
C.1.3 Reset Generator.....	120
<b>Index</b> .....	<b>122</b>
<b>Schematics</b> .....	<b>125</b>

# 1. INTRODUCTION

The RCM4300 series of RabbitCore modules is one of the next generation of core modules that take advantage of new Rabbit<sup>®</sup> 4000 features such as hardware DMA, clock speeds of up to 60 MHz, I/O lines shared with up to six serial ports and four levels of alternate pin functions that include variable-phase PWM, auxiliary I/O, quadrature decoder, and input capture. Coupled with more than 500 new opcode instructions that help to reduce code size and improve processing speed, this equates to a core module that is fast, efficient, and the ideal solution for a wide range of embedded applications. The RCM4300 also features an integrated 10/100Base-T Ethernet port, an optional A/D converter, and removable (“hot-swappable”) memory cards.

Each production model has a Development Kit with the essentials that you need to design your own microprocessor-based system, and includes a complete Dynamic C software development system. The Development Kits also contains a Prototyping Board that will allow you to evaluate the specific RCM4300 module and to prototype circuits that interface to the module. You will also be able to write and test software for the RCM4300 modules.

Throughout this manual, the term RCM4300 refers to the complete series of RCM4300 RabbitCore modules unless other production models are referred to specifically.

The RCM4300 has a Rabbit 4000 microprocessor operating at up to 58.98 MHz, a fast program-execution SRAM, data SRAM, serial flash memory, an 8-channel A/D converter, two clocks (main oscillator and timekeeping), and the circuitry necessary for reset and management of battery backup of the Rabbit 4000’s internal real-time clock and 512K of static RAM. One 50-pin header brings out the Rabbit 4000 I/O bus lines, parallel ports, A/D converter channels, and serial ports.

The RCM4300’s mass-storage capabilities make them suited to running the optional Dynamic C FAT file system module where data are stored and handled using the same directory file structure commonly used on PCs. A removable microSD Card can be hot-

swapped to transfer data quickly and easily using a standardized file system that can be read away from the RCM4300 installation.

The RCM4300 receives its +3.3 V power from the customer-supplied motherboard on which it is mounted. The RCM4300 can interface with all kinds of CMOS-compatible digital devices through the motherboard.

## 1.1 RCM4300 Features

- Small size: 1.84" × 2.85" × 0.84" (47 mm × 72 mm × 21 mm)
- Microprocessor: Rabbit 4000 running at up to 58.98 MHz
- Up to 28 or 36 general-purpose I/O lines configurable with up to four alternate functions
- 3.3 V I/O lines with low-power modes down to 2 kHz
- Up to six CMOS-compatible serial ports — four ports are configurable as a clocked serial ports (SPI), and one or two ports are configurable as SDLC/HDLC serial ports.
- Combinations of up to eight single-ended or four differential 12-bit analog inputs (RCM4300 only)
- Alternate I/O bus can be configured for 8 data lines and 6 address lines (shared with parallel I/O lines), I/O read/write
- Up to 2 MB flash memory, up to 1 MB fast program-execution SRAM, and 512K data SRAM
- Removable microSD Card memory that may be used with the standardized directory structure supported by the Dynamic C FAT File System module and may be read with a standard Windows SD Card reader
- Real-time clock
- Watchdog supervisor

There are two RCM4300 production models. Table 1 summarizes their main features.

**Table 1. RCM4300 Features**

Feature	RCM4300	RCM4310	RCM4320
Microprocessor	Rabbit <sup>®</sup> 4000 at 58.98 MHz		
Data SRAM	512 K		
Fast Program-Execution SRAM	1 MB	512 K	1 MB
Serial Flash Memory (program/data)	2 MB	1 MB	4 MB
Flash Memory (data storage)	microSD Card 128 MB–2 GB*		
A/D Converter	12 bits	—	



**Table 1. RCM4300 Features (continued)**

Feature	RCM4300	RCM4310	RCM4320
Serial Ports	5 shared high-speed, CMOS-compatible ports: <ul style="list-style-type: none"> <li>• all 5 configurable as asynchronous (with IrDA), 4 as clocked serial (SPI), and 1 as SDLC/HDLC</li> <li>• 1 clocked serial port shared with programming port</li> <li>• 1 clocked serial port shared with A/D converter, serial flash, and microSD Card</li> </ul>	6 shared high-speed, CMOS-compatible ports: <ul style="list-style-type: none"> <li>• all 6 configurable as asynchronous (with IrDA), 4 as clocked serial (SPI), and 2 as SDLC/HDLC</li> <li>• 1 clocked serial port shared with programming port</li> <li>• 1 clocked serial port shared with serial flash and microSD Card</li> </ul>	

\* MicroSD and microSDHC cards limited to four FAT16 partitions (2GB maximum) using short filenames. Dynamic C does not support FAT32 or long filenames. SDHC support added in Dynamic C 10.72C.

The RCM4300 is programmed over a standard PC USB port through a programming cable supplied with the Development Kit.

**NOTE:** The RabbitLink cannot be used to program RabbitCore modules based on the Rabbit 4000 microprocessor.

Appendix A provides detailed specifications for the RCM4300.

## 1.2 Advantages of the RCM4300

- Fast time to market using a fully engineered, “ready-to-run/ready-to-program” microprocessor core.
- Competitive pricing when compared with the alternative of purchasing and assembling individual components.
- Easy C-language program development and debugging.
- Rabbit Field Utility to download compiled Dynamic C .bin files, and cloning board options for 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, with royalty-free TCP/IP software.

## 1.3 Development and Evaluation Tools

### 1.3.1 RCM4300 Development Kit

The RCM4300 Development Kit contains the hardware essentials you will need to use the RCM4300 module. The items in the Development Kit and their use are as follows.

- RCM4300 module.
- Prototyping Board.
- 1 GB microSD Card with SD Card adapter.
- Universal AC adapter, 12 V DC, 1 A (includes Canada/Japan/U.S., Australia/N.Z., U.K., and European style plugs).
- USB programming cable with 10-pin header.
- Cat. 5 Ethernet crossover cable.
- 10-pin header to DB9 serial cable.
- *Getting Started* instructions.
- A bag of accessory parts for use on the Prototyping Board.
- *Rabbit 4000 Processor Easy Reference* poster.
- Registration card.

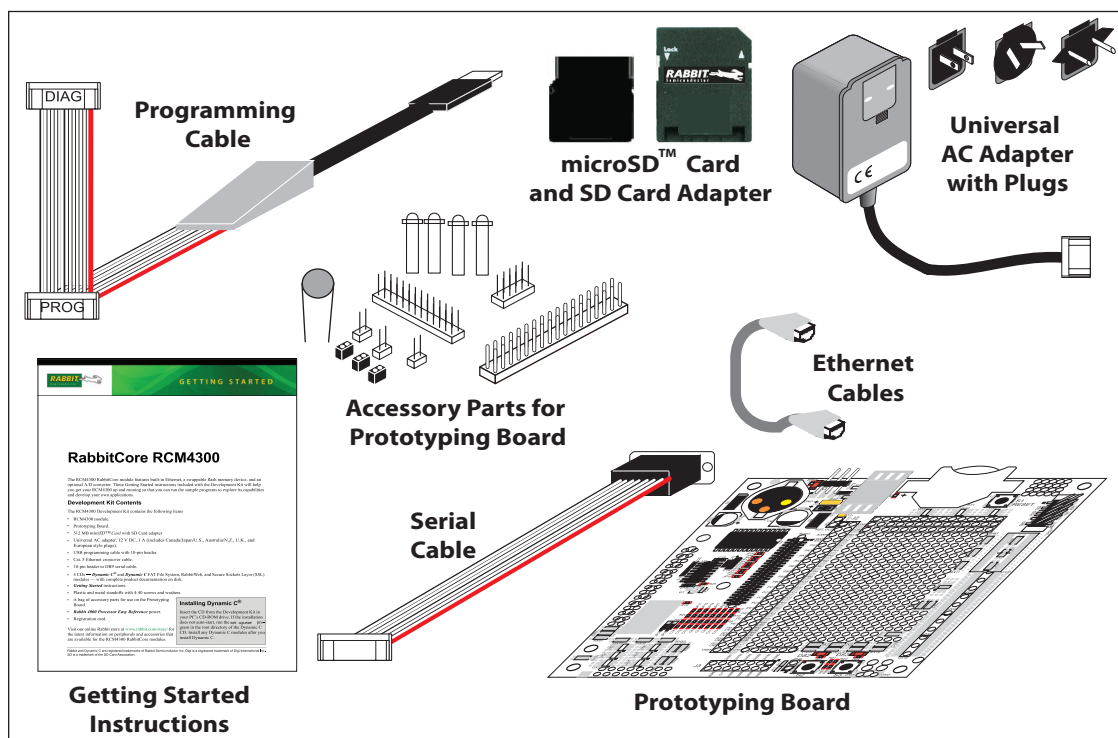


Figure 1. RCM4300 Development Kit

### 1.3.2 Software

The RCM4300 is programmed using version 10.21 or later of Dynamic C. You can download Dynamic C from the [support page](#) for this product.

You will also find the add-on Dynamic C modules containing the popular  $\mu$ C/OS-II real-time operating system, the FAT file system, as well as PPP, Advanced Encryption Standard (AES), and other select libraries. Digi offers multiple support levels to meet your needs. Visit [www.digi.com/support](http://www.digi.com/support) for more information.

### 1.3.3 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 available at [www.digi.com/support](http://www.digi.com/support).



## 2. GETTING STARTED

This chapter describes the RCM4300 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 RCM4300 Development Kit. If you purchased an RCM4300 module by itself, you will have to adapt the information in this chapter and elsewhere to your test and development setup.

### 2.1 Install Dynamic C

To develop and debug programs for the RCM4300 series of modules (and for all other Rabbit hardware), you must install and use Dynamic C.

If you have not yet installed Dynamic C version 10.21 (or a later version), download the software now from the [support page](#) for this product.

If autorun is disabled or the installation does not start, use the Windows **Start | Run** menu or Windows Disk Explorer to launch **setup.exe** from the root folder.

The installation program will guide you through the installation process. Most steps of the process are self-explanatory.

Once your installation is complete, you will have up to three new icons on your PC desktop. One icon is for Dynamic C, another opens the documentation menu, and the third is for the Rabbit Field Utility, a tool used to download precompiled software to a target system.

## 2.2 Hardware Connections

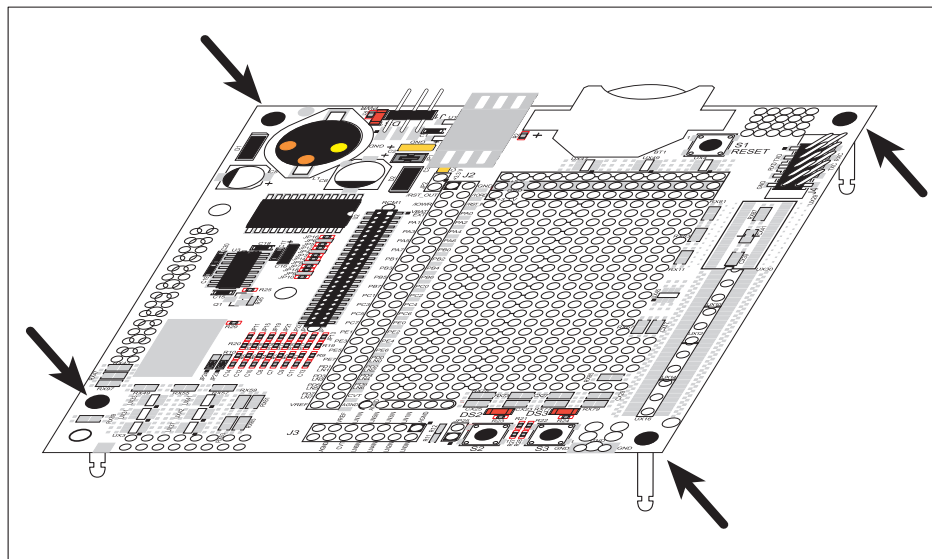
There are four steps to connecting the Prototyping Board for use with Dynamic C and the sample programs:

1. Prepare the Prototyping Board for Development.
2. Attach the RCM4300 module to the Prototyping Board.
3. Connect the programming cable between the RCM4300 and the PC.
4. Connect the power supply to the Prototyping Board.

### 2.2.1 Step 1 — Prepare the Prototyping Board for Development

Snap in four of the plastic standoffs supplied in the bag of accessory parts from the Development Kit in the holes at the corners as shown in Figure 2.

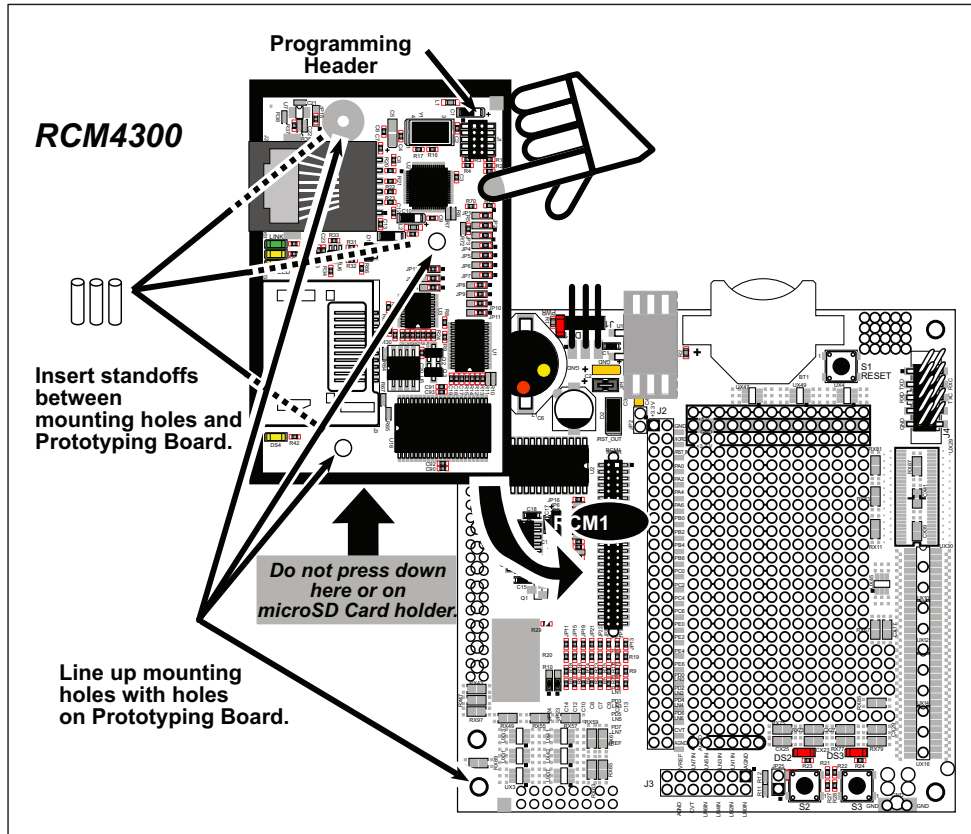
**NOTE:** Pay attention to use the hole that is pointed out towards the bottom left of the Prototyping Board since the hole below it is used for a standoff when mounting the RCM4300 on the Prototyping Board.



**Figure 2. Insert Standoffs**

## 2.2.2 Step 2 — Attach Module to Prototyping Board

Turn the RCM4300 module so that the mounting holes line up with the corresponding holes on the Prototyping Board. Insert the metal standoffs as shown in Figure 3, secure them from the bottom using the 4-40 × 3/16 screws and washers, then insert the module's header J4 on the bottom side into socket RCM1 on the Prototyping Board.



**Figure 3. Install the Module on the Prototyping Board**

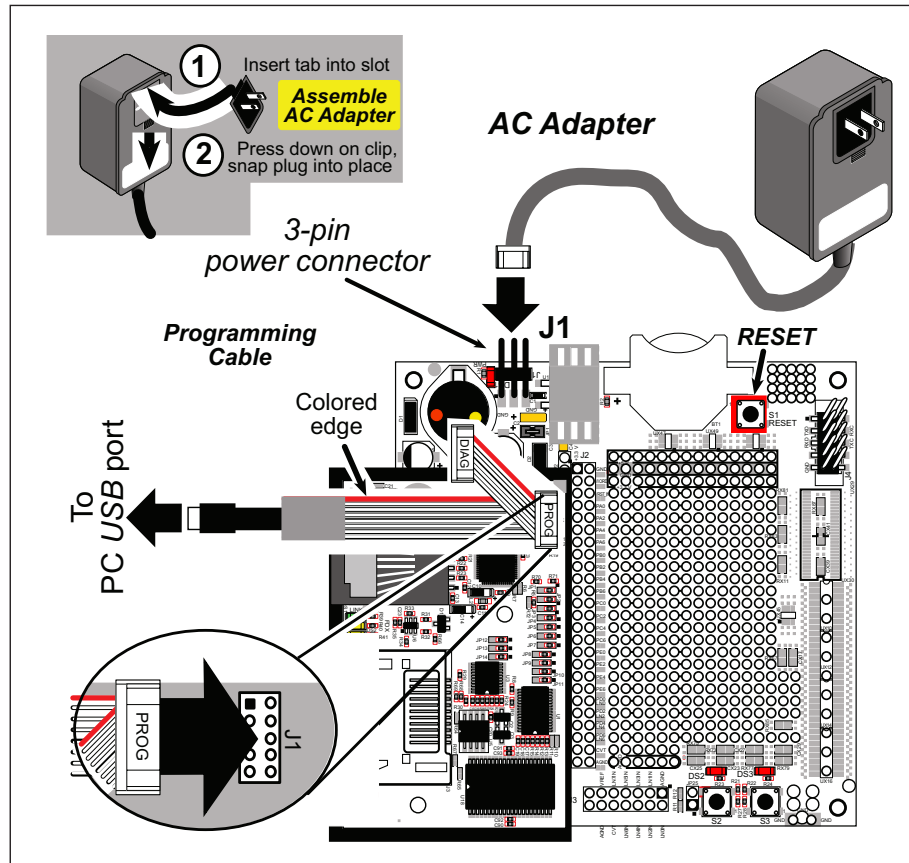
**NOTE:** It is important that you line up the pins on header J4 of the module exactly with socket RCM1 on the Prototyping Board. The header pins may become bent or damaged if the pin alignment is offset, and the module will not work. Permanent electrical damage to the module may also result if a misaligned module is powered up.

Press the module's pins gently into the Prototyping Board socket—press down in the area above the header pins. For additional integrity, you may secure the RCM4300 to the standoffs from the top using the remaining three screws and washers.

### 2.2.3 Step 3 — Connect Programming Cable

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

Connect the 10-pin connector of the programming cable labeled **PROG** to header J1 on the RCM4300 as shown in Figure 4. 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 presently not supported by the RCM4300.)



**Figure 4. Connect Programming Cable and Power Supply**

**NOTE:** Never disconnect the programming cable by pulling on the ribbon cable. Carefully pull on the connector to remove it from the header.

Connect the other end of the programming cable to an available USB port on your PC or workstation.

Your PC should recognize the new USB hardware, and the LEDs in the shrink-wrapped area of the USB programming cable will flash — if you get an error message, you will have to install USB drivers. Drivers for Windows XP are available in the Dynamic C **Drivers\Rabbit USB Programming Cable\WinXP\_2K** folder — double-click **DPInst.exe** to install the USB drivers. Drivers for other operating systems are available online at [www.ftdichip.com/Drivers/VCP.htm](http://www.ftdichip.com/Drivers/VCP.htm).

## 2.2.4 Step 4 — Connect Power

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

First, prepare the AC adapter for the country where it will be used by selecting the plug. The RCM4300 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 4, then press down on the spring-loaded clip below the plug assembly to allow the plug assembly to click into place. Release the clip to secure the plug assembly in the AC adapter.

Connect the AC adapter to 3-pin header J1 on the Prototyping Board as shown in Figure 4. The connector may be attached either way as long as it is not offset to one side—the center pin of J1 is always connected to the positive terminal, and either edge pin is ground.

Plug in the AC adapter. The **PWR** LED on the Prototyping Board next to the power connector at J1 should light up. The RCM4300 and the Prototyping Board are now ready to be used.

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

To power down the Prototyping Board, unplug the power connector from J1. You should disconnect power before making any circuit adjustments in the prototyping area, changing any connections to the board, or removing the RCM4300 from the Prototyping Board.



## 2.3 Run a Sample Program

Once the RCM4300 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. Select **Code and BIOS in RAM** on the “Compiler” tab in the Dynamic C **Options > Project Options** menu. Then click on the “Communications” tab and verify that **Use USB to Serial Converter** is selected to support the USB programming cable. Click **OK**.

**NOTE:** The **Code and BIOS in RAM** BIOS memory compiler option is recommended while debugging for faster download times. Remember to recompile the working application using the **Code and BIOS in Flash, Run in RAM** option once you are ready to use the RCM4300 in a standalone installation.

Determine which COM port was assigned to the USB programming cable on your PC. Open **Control Panel > System > Hardware > Device Manager > Ports** and identify which COM port is used for the USB connection. In Dynamic C, select **Options > Project Options**, then select this COM port on the **Communications** tab, then click **OK**. You may type the COM port number followed by **Enter** on your computer keyboard if the COM port number is outside the range on the dropdown menu.

Use the **File** menu to open the sample program **PONG.C**, which is in the Dynamic C **SAMPLES** folder. Press function key **F9** to compile and run the program. The **STUDIO** window will open on your PC and will display a small square bouncing around in a box.

This program shows that the CPU is working. The sample program described in Section 6.5, “Run the PINGME.C Sample Program,” tests the TCP/IP portion of the board.

### 2.3.1 Troubleshooting

If you receive the message **No Rabbit Processor Detected**, the programming cable may be connected to the wrong COM port, a connection may be faulty, or the target system may not be powered up. First, check to see that the power LED on the Prototyping Board is lit. If the LED is lit, check both ends of the programming cable to ensure that it is firmly plugged into the PC and the programming header on the RCM4300 with the marked (colored) edge of the programming cable towards pin 1 of the programming header. Ensure that the module is firmly and correctly installed in its connectors on the Prototyping Board.

If Dynamic C appears to compile the BIOS successfully, but you then receive a communication error message when you compile and load a 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 on the “Communications” tab in the Dynamic C **Options > Project Options** menu. Select a slower Max download baud rate. Click **OK** to save.

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 on the “Communications” tab in the Dynamic C **Options > Project Options** menu. Choose a lower debug baud rate. Click **OK** to save.

Press **<Ctrl-Y>** to force Dynamic C to recompile the BIOS. You should receive a **Bios compiled successfully** message once this step is completed successfully.

## 2.4 Where Do I Go From Here?

If the sample program ran fine, you are now ready to go on to the sample programs in Chapter 3 and to develop your own applications. The sample programs can be easily modified for your own use. The user's manual also provides complete hardware reference information and software function calls for the RCM4300 series of modules and the Prototyping Board.

For advanced development topics, refer to the *Dynamic C User's Manual*.

### 2.4.1 Technical Support

**NOTE:** If you purchased your RCM4300 through a distributor or through a Digi 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.
- Visit Digi's technical forum at [www.digi.com/support/forum](http://www.digi.com/support/forum).
- Visit Digi's Knowledge Base at [knowledge.digi.com](http://knowledge.digi.com).
- Contact Digi's Technical Support team at [tech.support@digi.com](mailto:tech.support@digi.com).

## 3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM4300 (and for all other Rabbit hardware), you must install and use Dynamic C. This chapter provides a tour of its major features with respect to the RCM4300.

### 3.1 Introduction

To help familiarize you with the RCM4300 modules, Dynamic C includes several sample programs. Loading, executing and studying these programs will give you a solid hands-on overview of the RCM4300's capabilities, as well as a quick start with Dynamic C as an application development tool.

**NOTE:** The sample programs assume 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.

In order to run the sample programs discussed in this chapter and elsewhere in this manual,

1. Your module must be plugged in to the Prototyping Board as described in Chapter 2, "Getting Started."
2. Dynamic C must be installed and running on your PC.
3. The programming cable must connect the programming header on the module to your PC.
4. Power must be applied to the module through the Prototyping Board.

Refer to Chapter 2, "Getting Started," if you need further information on these steps.

To run a sample program, open it with the **File** menu (if it is not still open), then compile and run it by pressing **F9**.

Each sample program has comments that describe the purpose and function of the program. Follow the instructions at the beginning of the sample program.

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

## 3.2 Sample Programs

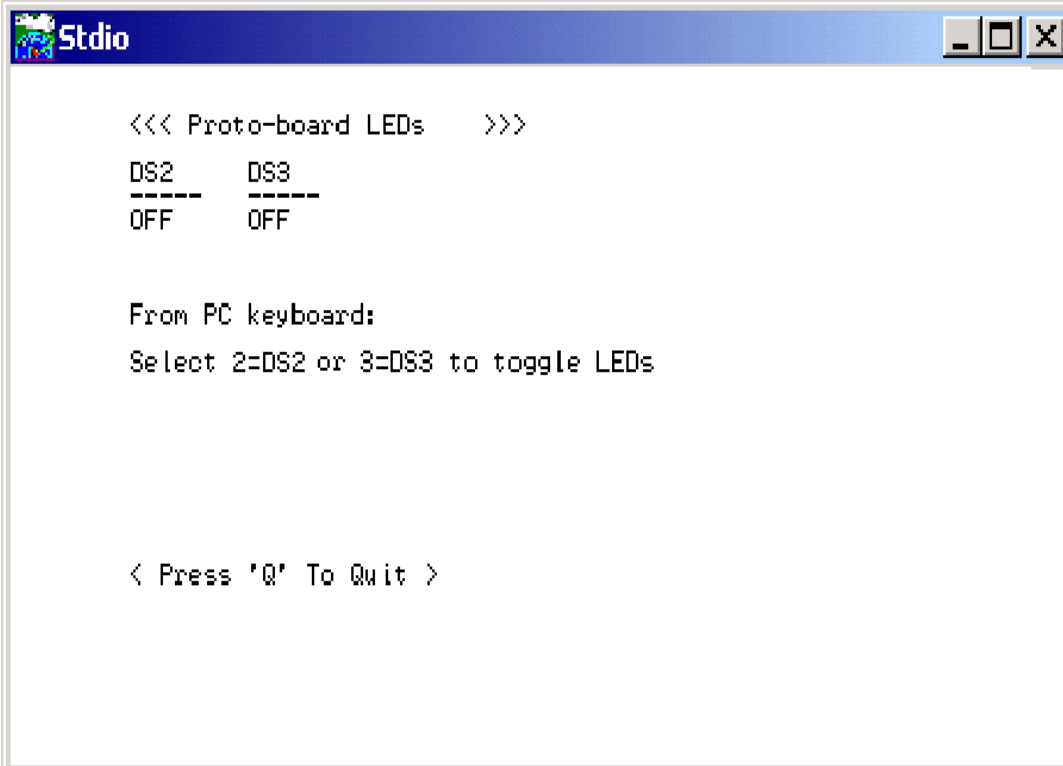
Of the many sample programs included with Dynamic C, several are specific to the RCM4300 modules. These programs will be found in the **SAMPLES\RCM4300** folder.

- **CONTROLLED.C**—Demonstrates use of the digital outputs by having you turn LEDs DS2 and DS3 on the Prototyping Board on or off from the **STDIO** window on your PC.

Parallel Port B bit 2 = LED DS2

Parallel Port B bit 3 = LED DS3

Once you compile and run **CONTROLLED.C**, the following display will appear in the Dynamic C **STDIO** window.



```
<<< Proto-board LEDs >>>
DS2  DS3
----  ----
OFF  OFF

From PC keyboard:
Select 2=DS2 or 3=DS3 to toggle LEDs

< Press 'Q' To Quit >
```

Press “2” or “3” on your keyboard to select LED DS2 or DS3 on the Prototyping Board. Then follow the prompt in the Dynamic C **STDIO** window to turn the LED ON or OFF. A logic low will light up the LED you selected.

- **FLASHLED1.C**—demonstrates the use of assembly language to flash LEDs DS2 and DS3 on the Prototyping Board at different rates. Once you have compiled and run this program, LEDs DS2 and DS3 will flash on/off at different rates.
- **FLASHLED2.C**—demonstrates the use of cofunctions and costatements to flash LEDs DS2 and DS3 on the Prototyping Board at different rates. Once you have compiled and run this program, LEDs DS2 and DS3 will flash on/off at different rates.

- **TOGGLESWITCH.C**—demonstrates the use of costatements to detect switch presses using the press-and-release method of debouncing. LEDs DS2 and DS3 on the Prototyping Board are turned on and off when you press switches S2 and S3. S2 and S3 are controlled by PB4 and PB5 respectively.

Once you have loaded and executed these five programs and have an understanding of how Dynamic C and the RCM4300 modules interact, you can move on and try the other sample programs, or begin building your own.

### 3.2.1 Tamper Detection

The tamper detection feature of the Rabbit 4000 microprocessor can be used to detect any attempt to enter the bootstrap mode. When such an attempt is detected, the VBAT RAM memory on the Rabbit 4000 chip is erased. The serial bootloader on RCM4300 Rabbit-Core modules uses the bootstrap mode to load the SRAM, which erases the VBAT RAM memory on any reset, and so it cannot be used for tamper detection. Therefore, no tamper detection sample program is available for RCM4300 RabbitCore modules.

### 3.2.2 Use of microSD Cards

The following sample program can be found in the **SAMPLES\RCM4300\SD\_Flash** folder.

- **SDFLASH\_INSPECT.C**—This program is a utility for inspecting the contents of a microSD Card. It provides examples of both reading and writing pages or sectors to the microSD Card. When the sample program starts running, it attempts to initialize the microSD Card on Serial Port B. The following five commands are displayed in the Dynamic C **STUDIO** window if a microSD Card is found:

- p — print out the contents of a specified page on the microSD Card
- r — print out the contents of a range of pages on the microSD Card
- c — clear (set to zero) all of the bytes in a specified page
- f — sets all bytes on the specified page to the given value
- t — write user-specified text to a selected page

The sample program prints out a single line for a page if all bytes in the page are set to the same value. Otherwise it prints a hex/ASCII dump of the page.

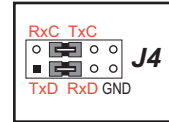
This utility works with the microSD Card at its lowest level, and writing to pages will likely make the microSD Card unreadable by a PC. For PC compatibility, you must use the Dynamic C FAT file system module, which allows you to work with files on the microSD Card in a way that they will be PC-compatible.

### 3.2.3 Serial Communication

The following sample programs are found in the **SAMPLES\RCM4300\SERIAL** folder.

- **FLOWCONTROL.C**—This program demonstrates how to configure Serial Port C for CTS/RTS flow control with serial data coming from Serial Port D (TxD) at 115,200 bps. The serial data received are displayed in the **STDIO** window.

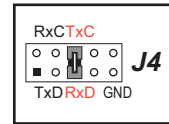
To set up the Prototyping Board, you will need to tie TxD and RxD together on the RS-232 header at J4, and you will also tie TxC and RxC together using the jumpers supplied in the Development Kit as shown in the diagram.



A repeating triangular pattern should print out in the **STDIO** window. The program will periodically switch flow control on or off to demonstrate the effect of flow control.

If you have two Prototyping Boards with modules, run this sample program on the sending board, then disconnect the programming cable and reset the sending board so that the module is operating in the Run mode. Connect TxC, TxD, and GND on the sending board to RxC, RxD, and GND on the other board using jumper wires, then, with the programming cable attached to the other module, run the sample program.

- **PARITY.C**—This program demonstrates the use of parity modes by repeatedly sending byte values 0–127 from Serial Port C to Serial Port D. The program will switch between generating parity or not on Serial Port C. Serial Port D will always be checking parity, so parity errors should occur during every other sequence.



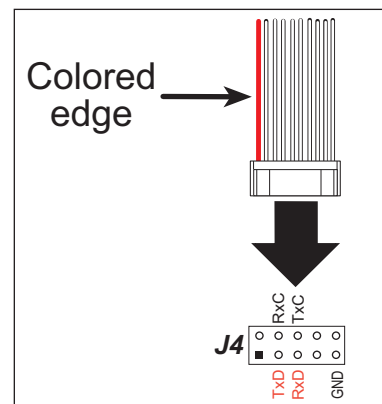
To set up the Prototyping Board, you will need to tie TxC and RxD together on the RS-232 header at J4 using one of the jumpers supplied in the Development Kit as shown in the diagram.

The Dynamic C **STDIO** window will display the error sequence.

- **SERDMA.C**—This program demonstrates using DMA to transfer data from a circular buffer to the serial port and vice versa. The Dynamic C **STDIO** window is used to view or clear the buffer.

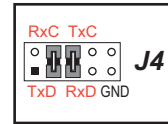
Before you compile and run the sample program, you will need to connect the RS-232 header at J4 to your PC as shown in the diagram using the serial to DB9 cable supplied in the Development Kit.

Once you have compiled and run the sample program, start Tera Term or another terminal emulation program to connect to the selected PC serial port at a baud rate of 115,200 bps. You can observe the output in the Dynamic C **STDIO** window as you type in Tera Term, and you can also use the Dynamic C **STDIO** window to clear the buffer.



The Tera Term utility can be downloaded from [hp.vector.co.jp/authors/VA002416/ter-aterm.html](http://hp.vector.co.jp/authors/VA002416/ter-aterm.html).

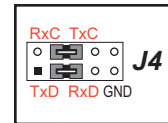
- **SIMPLE3WIRE.C**—This program demonstrates basic RS-232 serial communication. Lower case characters are sent on TxC, and are received by RxD. The received characters are converted to upper case and are sent out on TxD, are received on RxC, and are displayed in the Dynamic C **STDIO** window.



To set up the Prototyping Board, you will need to tie TxD and RxC together on the RS-232 header at J4, and you will also tie RxD and TxC together using the jumpers supplied in the Development Kit as shown in the diagram.

- **SIMPLE5WIRE.C**—This program demonstrates 5-wire RS-232 serial communication with flow control on Serial Port C and data flow on Serial Port D.

To set up the Prototyping Board, you will need to tie TxD and RxD together on the RS-232 header at J4, and you will also tie TxC and RxC together using the jumpers supplied in the Development Kit as shown in the diagram.

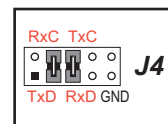


Once you have compiled and run this program, you can test flow control by disconnecting the TxC jumper from RxC while the program is running. Characters will no longer appear in the **STDIO** window, and will display again once TxC is connected back to RxC.

If you have two Prototyping Boards with modules, run this sample program on the sending board, then disconnect the programming cable and reset the sending board so that the module is operating in the Run mode. Connect TxC, TxD, and GND on the sending board to RxC, RxD, and GND on the other board using jumper wires, then, with the programming cable attached to the other module, run the sample program. Once you have compiled and run this program, you can test flow control by disconnecting TxC from RxC as before while the program is running. Since the J4 header locations on the two Prototyping Boards are connected with wires, there are no slip-on jumpers at J4 on either Prototyping Board.

- **SWITCHCHAR.C**—This program demonstrates transmitting and then receiving an ASCII string on Serial Ports C and D. It also displays the serial data received from both ports in the **STDIO** window.

To set up the Prototyping Board, you will need to tie TxD and RxC together on the RS-232 header at J4, and you will also tie RxD and TxC together using the jumpers supplied in the Development Kit as shown in the diagram.



Once you have compiled and run this program, press and release switches S2 and S3 on the Prototyping Board. The data sent between the serial ports will be displayed in the **STDIO** window.

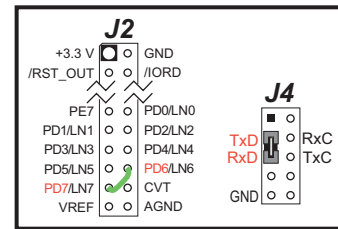
- **IOCONFIG\_SWITCHECHO.C**—This program demonstrates how to set up Serial Port E, which then transmits and then receives an ASCII string when switch S2 is pressed. The echoed serial data are displayed in the Dynamic C **STDIO** window.

Note that the I/O lines that carry the Serial Port E signals are not the Rabbit 4000 defaults. The Serial Port E I/O lines are configured by calling the library function **serEconfig()** that was generated by the Rabbit 4000 **IOCONFIG.EXE** utility program. Serial Port E is configured to use Parallel Port E bits PD6 and PD7. These signals are available on the Prototyping Board's Module Extension Header (header J2).

Serial Port D is left in its default configuration, using Parallel Port C bits PC0 and PC1. These signals are available on the Prototyping Board's RS-232 connector (header J4). Serial Port D transmits and then receives an ASCII string when switch S3 is pressed.

Also note that there is one library generated by **IOCONFIG.EXE** in the Dynamic C **SAMPLES\RCM4300\SERIAL** folder for the 29 MHz RCM4210.

To set up the Prototyping Board, you will need to tie TxD and RxD together on the RS-232 header at J4 using the jumpers supplied in the Development Kit; you will also tie TxE (PD6) and RxE (PD7) together with a soldered wire or with a wire jumper if you have soldered in the IDC header supplied with the accessory parts in the Development Kit.



Once you have compiled and run this program, press and release switches S2 or S3 on the Prototyping Board. The data echoed between the serial ports will be displayed in the **STDIO** window.



### 3.2.4 A/D Converter Inputs (RCM4300 only)

The following sample programs are found in the `SAMPLES\RCM4300\ADC` folder.

- **AD\_CAL\_ALL.C**—Demonstrates how to recalibrate all the single-ended analog input channels with one gain using two known voltages to generate the calibration constants for each channel. The constants will be written into the user block data area.

Connect a positive voltage from 0–20 V DC (for example, the power supply positive output) to analog input channels LN0IN–LN6IN on the Prototyping Board, and connect the ground to GND. Use a voltmeter to measure the voltage, and follow the instructions in the Dynamic C **STDIO** window once you compile and run this sample program. Remember that analog input LN7 on the Prototyping Board is used with the thermistor and is not be used with this sample program.

**NOTE:** The above sample program will overwrite the existing calibration constants.

- **AD\_CAL\_CHAN.C**—Demonstrates how to recalibrate one single-ended analog input channel with one gain using two known voltages to generate the calibration constants for that channel. The constants will be rewritten into the user block data area.

Connect a positive voltage from 0–20 V DC (for example, the power supply positive output) to an analog input channel on the Prototyping Board, and connect the ground to GND. Use a voltmeter to measure the voltage, and follow the instructions in the Dynamic C **STDIO** window once you compile and run this sample program. Remember that analog input LN7 on the Prototyping Board is used with the thermistor and is not be used with this sample program.

**NOTE:** The above sample program will overwrite the existing calibration constants for the selected channel.

- **AD\_RDVOLT\_ALL.C**—Demonstrates how to read all single-ended A/D input channels using previously defined calibration constants. The constants used to compute equivalent voltages are read from the user block data area, so the sample program cannot be run using the “Code and BIOS in RAM” compiler option.

Compile and run this sample program once you have connected a positive voltage from 0–20 V DC (for example, the power supply positive output) to analog input channels LN0IN–LN6IN on the Prototyping Board, and ground to GND. Follow the prompts in the Dynamic C **STDIO** window. Raw data and the computed equivalent voltages will be displayed. Remember that analog input LN7 on the Prototyping Board is used with the thermistor and is not be used with this sample program.

- **AD\_SAMPLE.C**—Demonstrates how to use a low level driver on single-ended inputs. The program will continuously display the voltage (averaged over 10 samples) that is present on an A/D converter channel (except LN7, which is reserved for use with a thermistor—see Appendix B.4.3.2). The constants used to compute equivalent voltages are read from the user block data area, so the sample program cannot be run using the “Code and BIOS in RAM” compiler option.

Compile and run this sample program once you have connected a positive voltage from 0–20 V DC to an analog input (except LN7) on the Prototyping Board, and ground to GND. Follow the prompts in the Dynamic C **STDIO** window. Raw data and the computed equivalent voltages will be displayed. If you attach a voltmeter between the

analog input and ground, you will be able to observe that the voltage in the Dynamic C **STDIO** window tracks the voltage applied to the analog input as you vary it.

- **THERMISTOR.C**—Demonstrates how to use analog input LN7 to calculate temperature for display to the Dynamic C **STDIO** window. This sample program assumes that the thermistor is the one included in the Development Kit whose values for beta, series resistance, and resistance at standard temperature are given in the part specification.

Install the thermistor at location JP25 on the Prototyping Board before running this sample program. Observe the temperature changes shown in the Dynamic C **STDIO** window as you apply heat or cold air to the thermistor.

### 3.2.4.1 Downloading and Uploading Calibration Constants

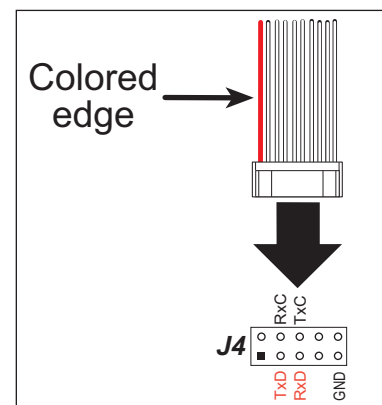
The Tera Term utility called for in these sample programs can be downloaded from <http://www.vector.co.jp/authors/VA002416/teraterm.html>.

These sample programs must be compiled to run from the fast SRAM. To do so, select **Options > Project Options** in Dynamic C, then select the “Compiler” tab, and select “**Code and BIOS in Flash, Run in RAM**” for the **BIOS Memory Setting**.

Before you compile and run these sample programs, you will also need to connect the RS-232 header at J4 to your PC as shown in the diagram using the serial to DB9 cable supplied in the Development Kit.

- **DNLOADCALIB.C**—Demonstrates how to retrieve analog calibration data to rewrite it back to the user block using a terminal emulation utility such as Tera Term.

Start Tera Term or another terminal emulation program on your PC, and configure the serial parameters as follows.



- Baud rate 19,200 bps, 8 bits, no parity, 1 stop bit
- Enable **Local Echo** option
- Feed options — **Receive = CR, Transmit = CR + LF**

Now compile and run this sample program. Verify that the message “Waiting, Please Send Data file” message is being displayed in the Tera Term display window before proceeding.

Within Tera Term, select **File-->Send File-->Path and filename**, then select the **OPEN** option within the dialog box. Once the data file has been downloaded, Tera Term will indicate whether the calibration data were written successfully.

- **UPLOADCALIB.C**—Demonstrates how to read the analog calibration constants from the user block using a terminal emulation utility such as Tera Term.