



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!



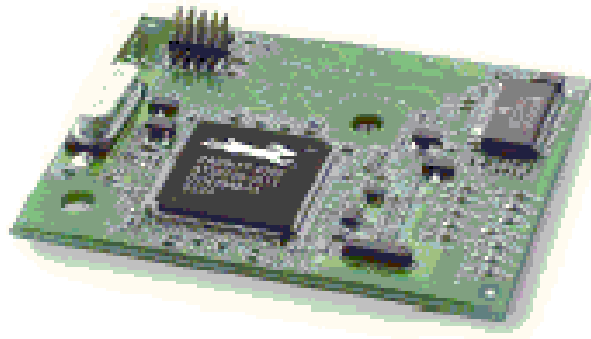
Contact us

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 RCM4100

C-Programmable Core Module

User's Manual

019-0153 • 090508-G

RabbitCore RCM4100 User's Manual

Part Number 019-0153 • 090508–G • Printed in U.S.A.

©2006–2009 Digi International Inc. • All rights reserved.

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

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

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

Trademarks

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

Rabbit 4000 and RabbitCore are trademarks of Digi International Inc.

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

Digi International Inc.

www.rabbit.com

TABLE OF CONTENTS

Chapter 1. Introduction	1
1.1 RCM4100 Features	2
1.2 Advantages of the RCM4100	4
1.3 Development and Evaluation Tools.....	5
1.3.1 RCM4110 Development Kit	5
1.3.2 RCM4100 Analog Development Kit	6
1.3.3 Software	6
1.3.4 Online Documentation	6
Chapter 2. Getting Started	7
2.1 Install Dynamic C	7
2.2 Hardware Connections.....	8
2.2.1 Step 1 — Prepare the Prototyping Board.....	8
2.2.2 Step 2 — Attach Module to Prototyping Board.....	9
2.2.3 Step 3 — Connect Programming Cable.....	10
2.2.4 Step 4 — Connect Power	11
2.3 Run a Sample Program	12
2.3.1 Troubleshooting	12
2.4 Where Do I Go From Here?	13
2.4.1 Technical Support	13
Chapter 3. Running Sample Programs	15
3.1 Introduction.....	15
3.2 Sample Programs	16
3.2.1 Serial Communication.....	18
3.2.2 A/D Converter Inputs (RCM4100 only)	21
3.2.2.1 Downloading and Uploading Calibration Constants.....	22
3.2.3 Real-Time Clock	24
Chapter 4. Hardware Reference	25
4.1 RCM4100 Digital Inputs and Outputs	26
4.1.1 Memory I/O Interface	32
4.1.2 Other Inputs and Outputs	32
4.2 Serial Communication	33
4.2.1 Serial Ports	33
4.2.1.1 Using the Serial Ports.....	34
4.2.2 Programming Port	35
4.3 Programming Cable	36
4.3.1 Changing Between Program Mode and Run Mode	36
4.3.2 Standalone Operation of the RCM4100.....	37
4.4 A/D Converter (RCM4100 only).....	38
4.4.1 A/D Converter Power Supply	40

4.5 Other Hardware	41
4.5.1 Clock Doubler	41
4.5.2 Spectrum Spreader.....	41
4.6 Memory	42
4.6.1 SRAM.....	42
4.6.2 Flash EPROM.....	42
Chapter 5. Software Reference	43
5.1 More About Dynamic C.....	43
5.2 Dynamic C Function Calls	45
5.2.1 Digital I/O.....	45
5.2.2 Serial Communication Drivers	45
5.2.3 SRAM Use.....	45
5.2.4 Prototyping Board Function Calls	47
5.2.4.1 Board Initialization.....	47
5.2.4.2 Alerts.....	48
5.2.5 Analog Inputs (RCM4100 only).....	49
5.3 Upgrading Dynamic C	66
5.3.1 Add-On Modules	66
Appendix A. RCM4100 Specifications	67
A.1 Electrical and Mechanical Characteristics	68
A.1.1 A/D Converter	72
A.1.2 Headers	73
A.2 Rabbit 4000 DC Characteristics	74
A.3 I/O Buffer Sourcing and Sinking Limit.....	75
A.4 Bus Loading	75
A.5 Jumper Configurations	78
A.6 Conformal Coating.....	80
Appendix B. Prototyping Board	81
B.1 Introduction	82
B.1.1 Prototyping Board Features	83
B.2 Mechanical Dimensions and Layout	85
B.3 Power Supply.....	86
B.4 Using the Prototyping Board.....	87
B.4.1 Adding Other Components	89
B.4.2 Measuring Current Draw	89
B.4.3 Analog Features (RCM4100 only)	90
B.4.3.1 A/D Converter Inputs.....	90
B.4.3.2 Thermistor Input	92
B.4.3.3 A/D Converter Calibration.....	92
B.4.4 Serial Communication	93
B.4.4.1 RS-232	94
B.5 Prototyping Board Jumper Configurations.....	95
Appendix C. Power Supply	99
C.1 Power Supplies	99
C.1.1 Battery Backup	99
C.1.2 Battery-Backup Circuit.....	100
C.1.3 Reset Generator.....	100
Index	103
Schematics	107



1. INTRODUCTION

The RCM4100 series is the first of the next-generation core modules that take advantage of new Rabbit[®] 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 Development Kit has the essentials that you need to design your own microprocessor-based system, and includes a complete Dynamic C software development system. This Development Kit also contains a Prototyping Board that will allow you to evaluate the RCM4100 series and to prototype circuits that interface to the RCM4100 series of modules. You will also be able to write and test software for these modules.

Throughout this manual, the term RCM4100 series refers to the complete series of RCM4100 RabbitCore modules unless other production models are referred to specifically.

The RCM4100 has a Rabbit 4000 microprocessor operating at up to 58.98 MHz, static RAM, flash memory, an 8-channel A/D converter, two clocks (main oscillator and time-keeping), and the circuitry necessary for reset and management of battery backup of the Rabbit 4000's internal real-time clock and the static RAM. One 50-pin header brings out the Rabbit 4000 I/O bus lines, parallel ports, and serial ports.

The RCM4100 series receives its +3.3 V power from the customer-supplied motherboard on which it is mounted. The RCM4100 series can interface with all kinds of CMOS-compatible digital devices through the motherboard.

1.1 RCM4100 Features

- Small size: 1.41" × 1.88" × 0.49"
(36 mm × 48 mm × 12 mm)
- Microprocessor: Rabbit 4000 running at up to 58.98 MHz
- Up to 40 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
- Six CMOS-compatible serial ports — four ports are configurable as a clocked serial port (SPI), and two ports are configurable as SDLC/HDLC serial ports.
- 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
- 512K flash memory, 256K data SRAM
- Real-time clock
- Watchdog supervisor

There are three RCM4100 production models. Table 1 summarizes their main features.

Table 1. RCM4100 Features

Feature	RCM4100	RCM4110	RCM4120
Microprocessor	Rabbit® 4000 at 58.98 MHz	Rabbit® 4000 at 29.49 MHz	Rabbit® 4000 at 58.98 MHz
Flash Memory	512K		
Data SRAM	512K	256K	512K
Fast Program-Execution SRAM	512K	—	512K
A/D Converter	12 bits	—	—
Serial Ports	6 high-speed, CMOS-compatible ports: <ul style="list-style-type: none"> • all 6 configurable as asynchronous (with IrDA), 4 as clocked serial (SPI), and 2 as SDLC/HDLC • 1 asynchronous clocked serial port shared with programming port • 1 clocked serial port shared with A/D converter 	6 high-speed, CMOS-compatible ports: <ul style="list-style-type: none"> • all 6 configurable as asynchronous (with IrDA), 4 as clocked serial (SPI), and 2 as SDLC/HDLC • 1 asynchronous clocked serial port shared with programming port 	

The RCM4100 series 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 RCM4100 series.

1.2 Advantages of the RCM4100

- Fast time to market using a fully engineered, “ready-to-run/ready-to-program” micro-processor 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.

1.3 Development and Evaluation Tools

1.3.1 RCM4110 Development Kit

The RCM4110 Development Kit contains the hardware essentials you will need to use your RCM4110 module. The items in the Development Kit and their use are as follows.

- RCM4110 module.
- Prototyping Board.
- Universal AC adapter, 12 V DC, 1 A (includes Canada/Japan/U.S., Australia/N.Z., U.K., and European style plugs). Development Kits sold in North America may contain an AC adapter with only a North American style plug.
- Programming cable with integrated level-matching circuitry.
- 10-pin header to DB9 serial cable.
- *Dynamic C*[®] CD-ROM, with complete product documentation on disk.
- *Getting Started* instructions.
- A bag of accessory parts for use on the Prototyping Board.
- *Rabbit 4000 Processor Easy Reference* poster.
- Registration card.

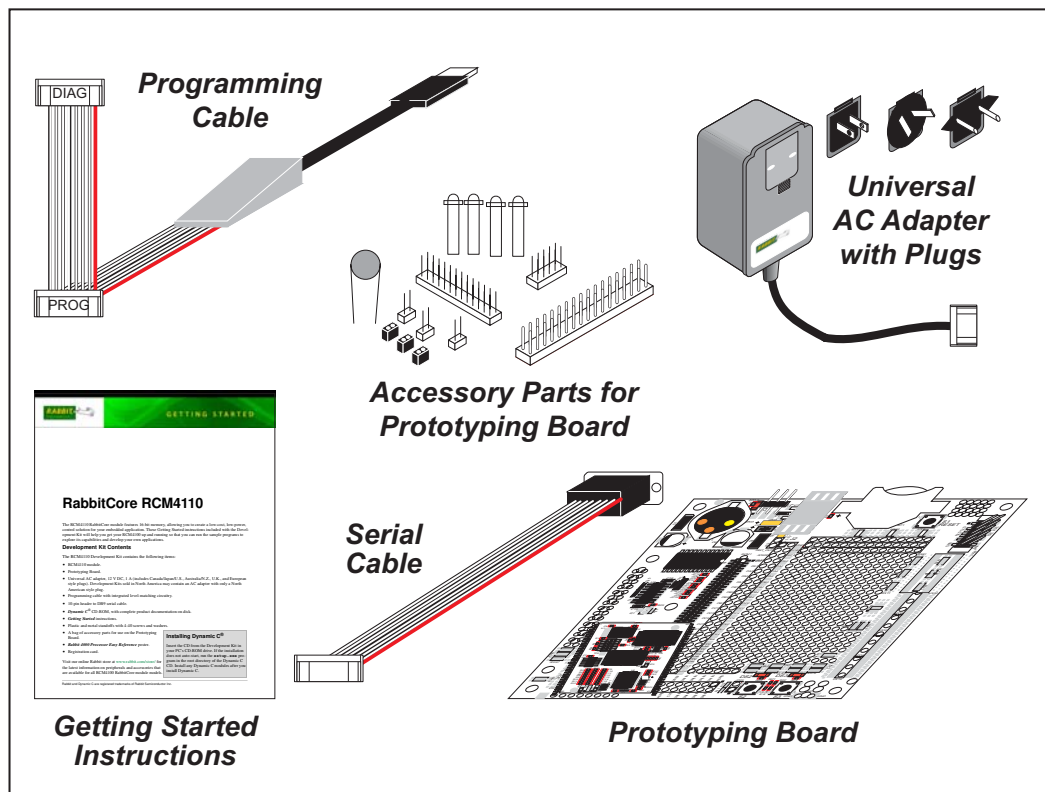


Figure 1. RCM4110 Development Kit

1.3.2 RCM4100 Analog Development Kit

The RCM4100 Analog Development Kit contains the hardware essentials you will need to use the RCM4100 module. The RCM4100 Analog Development Kit contents are similar to those of the RCM4110 Development Kit, except that the RCM4100 module is included instead of the RCM4110 module.

1.3.3 Software

The RCM4100 series is programmed using version 10.01 or later of Dynamic C. A compatible version is included on the Development Kit CD-ROM.

Starting with Dynamic C version 10.40, Dynamic C includes the popular μ C/OS-II real-time operating system, point-to-point protocol (PPP), FAT file system, RabbitWeb, and other select libraries. Rabbit also offers for purchase the Rabbit Embedded Security Pack featuring the Secure Sockets Layer (SSL) and a specific Advanced Encryption Standard (AES) library.

In addition to the Web-based technical support included at no extra charge, a one-year telephone-based technical support module is also available for purchase. Visit our Web site at www.rabbit.com or contact your Rabbit sales representative or authorized distributor for further information.

1.3.4 Online Documentation

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

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

2. GETTING STARTED

This chapter describes the RCM4100 series 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 RCM4100 Development Kit. If you purchased an RCM4100 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 RCM4100 series (and for all other Rabbit hardware), you must install and use Dynamic C.

If you have not yet installed Dynamic C version 10.01 (or a later version), do so now by inserting the Dynamic C CD from the RCM4100 Development Kit in your PC's CD-ROM drive. If autorun is enabled, the CD installation will begin automatically.

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 of the CD-ROM.

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

Dynamic C uses a COM (serial) port to communicate with the target development system. The installation allows you to choose the COM port that will be used. The default selection is COM1. You may select any available port for Dynamic C's use. If you are not certain which port is available, select COM1. This selection can be changed later within Dynamic C.

NOTE: The installation utility does not check the selected COM port in any way. Specifying a port in use by another device (mouse, modem, etc.) may lead to a message such as **"could not open serial port"** when Dynamic C is started.

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

If you have purchased any of the optional Dynamic C modules, install them after installing Dynamic C. The modules may be installed in any order. You must install the modules in the same directory where Dynamic C was installed.

2.2 Hardware Connections

There are three 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 RCM4100 module to the Prototyping Board.
3. Connect the programming cable between the RCM4100 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.

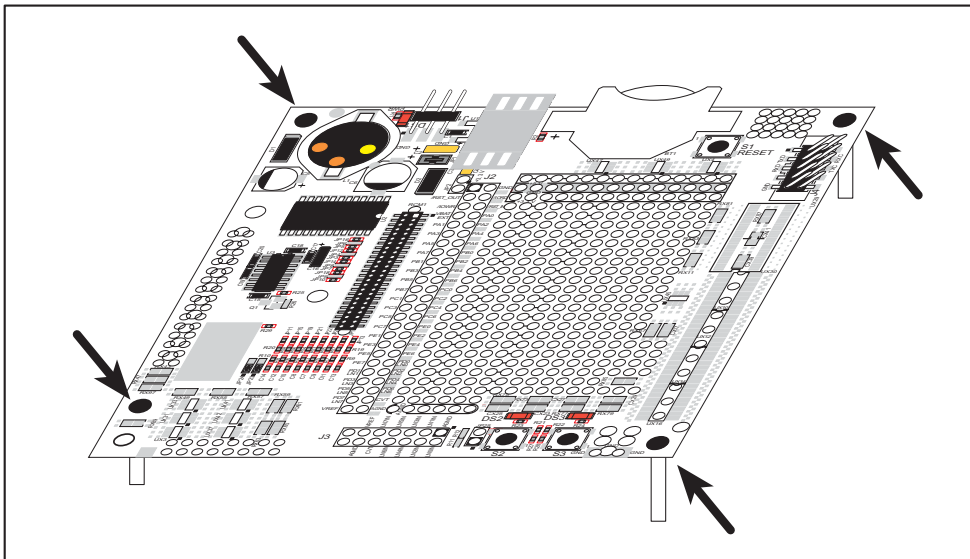


Figure 2. Insert Standoffs

2.2.2 Step 2 — Attach Module to Prototyping Board

Turn the RCM4100 module so that the mounting holes line up with the corresponding holes on the Prototyping Board. Insert the metal standoffs as shown, secure them from the bottom using two screws and washers, then insert the module's header J2 on the bottom side into socket RCM1 on the Prototyping Board.

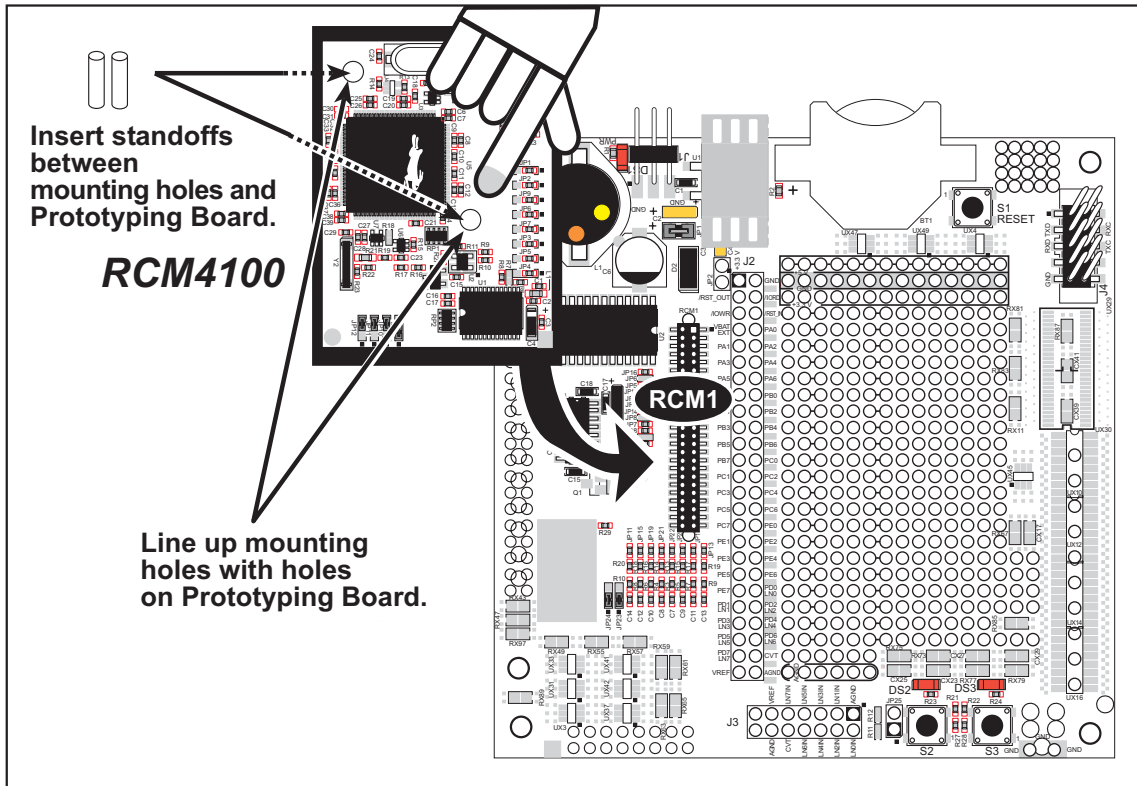


Figure 3. Install the Module on the Prototyping Board

NOTE: It is important that you line up the pins on header J2 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 RCM4100 to the standoffs from the top using the remaining two 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 RCM4100 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 used for a normal serial connection.)

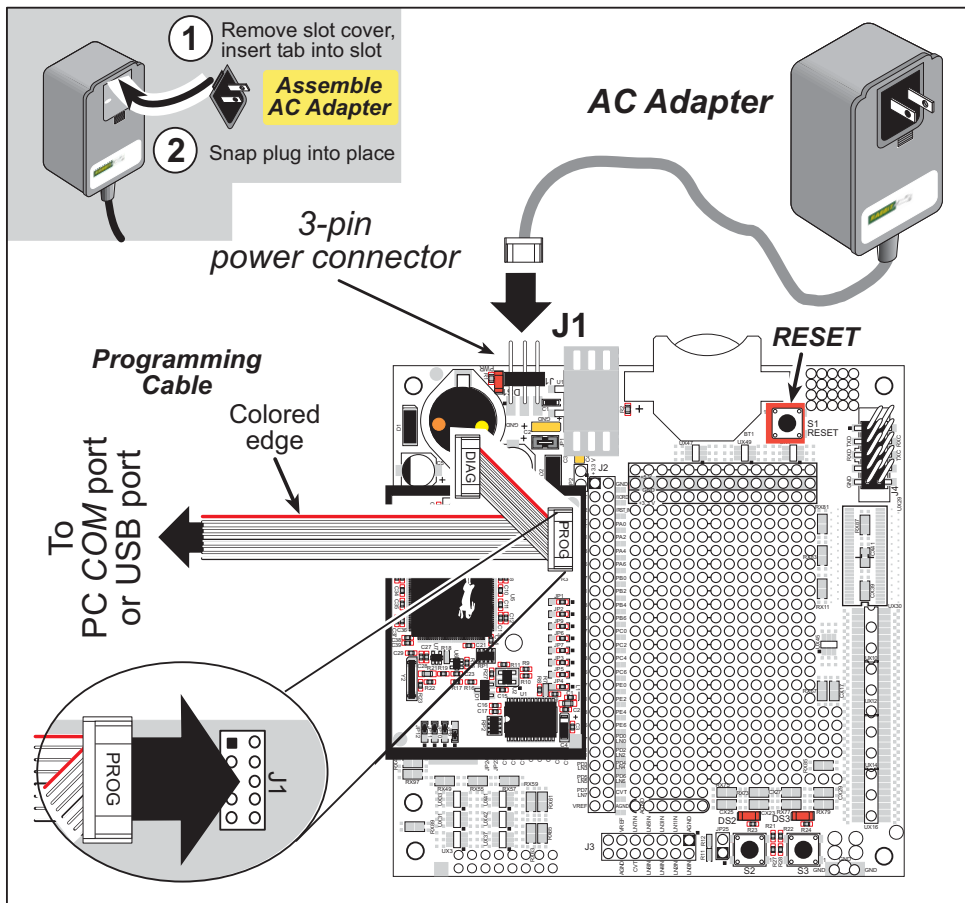


Figure 4. Connect Programming Cable and Power Supply

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

NOTE: Either a serial or a USB programming cable was supplied with this Development Kit. If you have a serial programming cable, an RS-232/USB converter (Rabbit Part No. 20-151-0178) is available to allow you to use the serial programming cable with a USB port.

Depending on the programming cable, connect the other end to a COM port or a USB port on your PC.

If you are using a USB programming cable, your PC should recognize the new USB hardware, and the LEDs in the shrink-wrapped area of the 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.

2.2.4 Step 4 — Connect Power

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

If you have the universal AC adapter, prepare the AC adapter for the country where it will be used by selecting the appropriate plug. Insert 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 plug until it clicks into place.

Connect the AC adapter to 3-pin header J1 on the Prototyping Board as shown in Figure 4 above. 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 RCM4100 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 RCM4100 from the Prototyping Board.

2.3 Run a Sample Program

Once the RCM4100/RCM4110 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.

If you are using a USB port to connect your computer to the RCM4100/RCM4110, click on the “Communications” tab and verify that **Use USB to Serial Converter** is selected to support the USB programming cable. Click **OK**. You may have to determine which COM port was assigned to the RS-232/USB converter. 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 drop-down menu.

Now find the file **PONG.C**, which is in the Dynamic C **SAMPLES** folder. To run the program, open it with the **File** menu, compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The **STDIO** window will open on your PC and will display a small square bouncing around in a box.

2.3.1 Troubleshooting

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 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 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 and that the jumper across pins 5–6 of header JP10 on the Prototyping Board is installed. 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 RCM4100 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 there are no faults with the hardware, select a different COM port within Dynamic C as explained for the USB port above. 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 for another available COM port. 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 the *RCM4100 User's Manual* (click the documentation icon on your PC) 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 RCM4100 and the Prototyping Board.

For advanced development topics, refer to the *Dynamic C User's Manual*, also in the online documentation set, which is on the Dynamic C CD in a `docs` folder..

2.4.1 Technical Support

NOTE: If you purchased your RCM4100 through a distributor or through a Rabbit partner, contact the distributor or partner first for technical support.

If there are any problems at this point:

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

3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM4100 series (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 RCM4100 series.

3.1 Introduction

To help familiarize you with the RCM4100 series of modules, Dynamic C includes several sample programs. Loading, executing and studying these programs will give you a solid hands-on overview of the RCM4100 series' 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.

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

3.2 Sample Programs

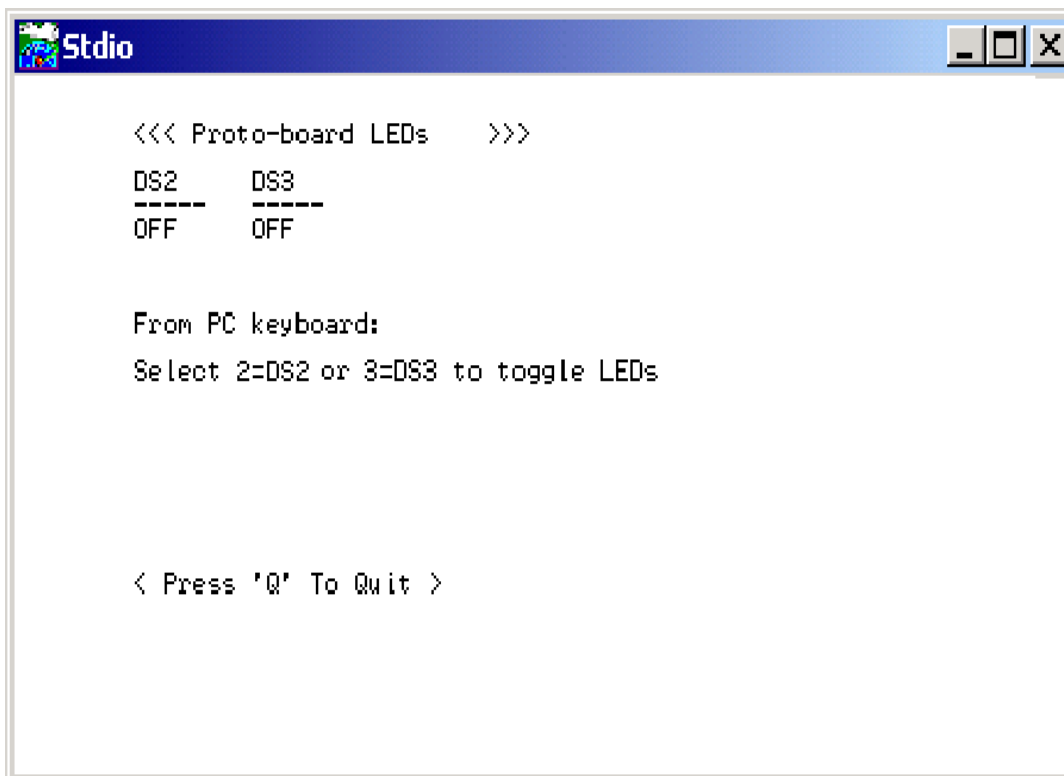
Of the many sample programs included with Dynamic C, several are specific to the RCM4100 series of modules. These programs will be found in the **SAMPLES\RCM4100** folder.

- **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.

- **LOW_POWER.C**—demonstrates how to implement a function in RAM to reduce power consumption by the Rabbit microprocessor. There are four features that lead to the lowest possible power draw by the microprocessor.
 1. Run the CPU from the 32 kHz crystal.
 2. Turn off the high-frequency crystal oscillator.
 3. Run from RAM.
 4. Ensure that internal I/O instructions do not use CS0.

Once you are ready to compile and run this sample program, use **<Alt-F9>** instead of just **F9**. This will disable polling, which will allow Dynamic C to continue debugging once the target starts running off the 32 kHz oscillator.

This sample program will toggle LEDs DS2 and DS3 on the Prototyping Board. You may use an oscilloscope. DS2 will blink the fastest. After switching to low power, both LEDs will blink together.

- **TAMPERDETECTION.C**—demonstrates how to detect an attempt to enter the bootstrap mode. When an attempt is detected, the battery-backed onchip-encryption RAM on the Rabbit 4000 is erased. This battery-backed onchip-encryption RAM can be useful to store data such as an AES encryption key from a remote location.

This sample program shows how to load and read the battery-backed onchip-encryption RAM and how to enable a visual indicator.

Once this sample is compiled running (you have pressed the **F9** key while the sample program is open), remove the programming cable and press the reset button on the Prototyping Board to reset the module. LEDs DS2 and DS3 will be flashing on and off.

Now press switch S2 to load the battery-backed RAM with the encryption key. The LEDs are now on continuously. Notice that the LEDs will stay on even when you press the reset button on the Prototyping Board.

Reconnect the programming cable briefly and unplug it again. The LEDs will be flashing because the battery-backed onchip-encryption RAM has been erased. Notice that the LEDs will continue flashing even when you press the reset button on the Prototyping Board.

You may press switch S2 again and repeat the last steps to watch the LEDs.

- **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.

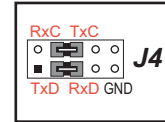
Once you have loaded and executed these five programs and have an understanding of how Dynamic C and the RCM4100 series of modules interact, you can move on and try the other sample programs, or begin building your own.

3.2.1 Serial Communication

The following sample programs are found in the `SAMPLES\RCM4100\SERIAL` folder.

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

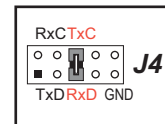
To set up the Prototyping Board, you will need to tie TxD and RxD together on the RS-232 header at J4, and you will also tie TxC and RxC together using the jumpers supplied in the Development Kit as shown in the diagram.



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 no 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, 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.

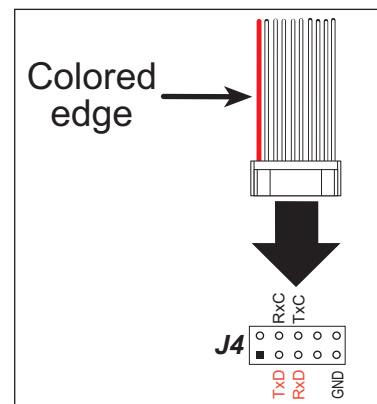
The Dynamic C **STDIO** window will display the error sequence.

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

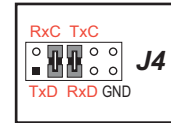
Before you compile and run the sample program, you will need to connect the RS-232 header at J4 to your PC as shown in the diagram using the serial to DB9 cable supplied in the Development Kit.

Once you have compiled and run the sample program, start Tera Term or another terminal emulation program to connect to the PC serial port using 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 serial utility can be downloaded from hp.vector.co.jp/authors/VA002416/teraterm.html.



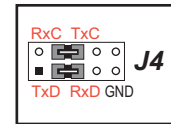
- **SIMPLE3WIRE.C**—This program demonstrates basic RS-232 serial communication. Lower case characters are sent by TxC, and are received by RxD. The characters are converted to upper case and are sent out by TxD, are received by RxC, and are displayed in the Dynamic C **STDIO** window.



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 D and data flow on Serial Port C.

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.

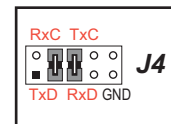


Once you have compiled and run this program, you can test flow control by disconnecting TxD from RxD while the program is running. Characters will no longer appear in the **STDIO** window, and will display again once TxD is connected back to RxD.

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, 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 TxD from RxD as before while the program is running.

- **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 Ports E and F, which then transmit and then receive an ASCII string when switch S2 or S3 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 and F signals are not the Rabbit 4000 defaults. The Serial Port E and F I/O lines are configured by calling the library function `serEFconfig()` that was generated by the Rabbit 4000 **IOCONFIG.EXE** utility program.

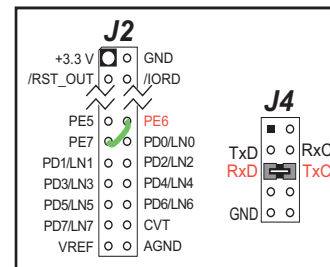
Serial Port E is configured to use Parallel Port E bits PE6 and PE7. These signals are available on the Prototyping Board's Module Extension Header (header J2).

Serial Port F is configured to use Parallel Port C bits PC2 and PC3. These signals are available on the Prototyping Board's RS-232 connector (header J4).

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 with Serial Port F when switch S3 is pressed.

Also note that there are two libraries generated by **IOCONFIG.EXE** in the Dynamic C **SAMPLES\RCM4100\SERIAL** folder for the 29 MHz RCM4110 and the 58 MHz RCM4100 and RCM4120.

To set up the Prototyping Board, you will need to tie TxD and Rx C together on the RS-232 header at J4 using the jumpers supplied in the Development Kit; you will also tie Tx E (PD6) and Rx E (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.

3.2.2 A/D Converter Inputs (RCM4100 only)

The following sample programs are found in the `SAMPLES\RCM4100\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). 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.