Welcome to E-XFL.COM

Understanding Embedded - Microcontroller, Microprocessor, FPGA Modules

Embedded - Microcontroller, Microprocessor, and FPGA Modules are fundamental components in modern electronic systems, offering a wide range of functionalities and capabilities. Microcontrollers are compact integrated circuits designed to execute specific control tasks within an embedded system. They typically include a processor, memory, and input/output peripherals on a single chip. Microprocessors, on the other hand, are more powerful processing units used in complex computing tasks, often requiring external memory and peripherals. FPGAs (Field Programmable Gate Arrays) are highly flexible devices that can be configured by the user to perform specific logic functions, making them invaluable in applications requiring customization and adaptability.

Applications of Embedded - Microcontroller,

Details	
Product Status	Obsolete
Module/Board Type	MPU Core
Core Processor	Rabbit 4000
Co-Processor	-
Speed	58.98MHz
Flash Size	1MB (Internal), 32MB (External)
RAM Size	1MB
Connector Type	IDC Header 2x25, 2x5
Size / Dimension	1.84" x 2.42" (47mm x 61mm)
Operating Temperature	0°C ~ 70°C
Purchase URL	https://www.e-xfl.com/product-detail/digi-international/20-101-1215

1.1 RCM4000 Features

- Small size: 1.84" × 2.42" × 0.77" (47 mm × 61 mm × 20 mm)
- Microprocessor: Rabbit 4000 running at 58.98 MHz
- Up to 29 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 five CMOS-compatible serial ports — four ports are configurable as a clocked serial ports (SPI), and one port is configurable as an SDLC/HDLC serial port.
- Combinations of up to eight single-ended or four differential 12-bit analog inputs (RCM4000 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
- 512KB or 1MB flash memory, 512KB or 1 MB SRAM, with a fixed mass-storage flash-memory option that may be used with the standardized directory structure supported by the Dynamic C FAT File System module
- Real-time clock
- Watchdog supervisor

There are three RCM4000 production models. Table 1 summarizes their main features.

Table 1. RCM4000 Features

Feature	RCM4000	RCM4010	RCM4050
Microprocessor	Rabbit® 4000 at 58.98 MHz		
SRAM	512KB		1MB
Flash Memory (program)	512KB		1MB
Flash Memory (mass data storage)	32 MB (NAND flash)	—	32 MB (NAND flash)
A/D Converter	12 bits	—	—
Serial Ports	4 shared high-speed, CMOS-compatible ports: • all 4 configurable as asynchronous (with IrDA) or as clocked serial (SPI) • 1 asynchronous clocked serial port shared with programming port • 1 clocked serial port shared with A/D converter	5 shared high-speed, CMOS-compatible ports: • all 5 configurable as asynchronous (with IrDA), 4 as clocked serial (SPI), and 1 as SDLC/HDLC • 1 asynchronous clocked serial port shared with programming port	4 shared high-speed, CMOS-compatible ports: • all 4 configurable as asynchronous (with IrDA) or as clocked serial (SPI) • 1 asynchronous clocked serial port shared with programming port

2. GETTING STARTED

This chapter describes the RCM4000 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 RCM4000 Analog or the RCM4010 Development Kit. If you purchased an RCM4000 or RCM4010 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 RCM4000 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 10.03 (or a later version), do so now by inserting the Dynamic C CD from the 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 RCM4000 or RCM4010 module to the Prototyping Board.
3. Connect the programming cable between the RCM4000 or RCM4010 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.

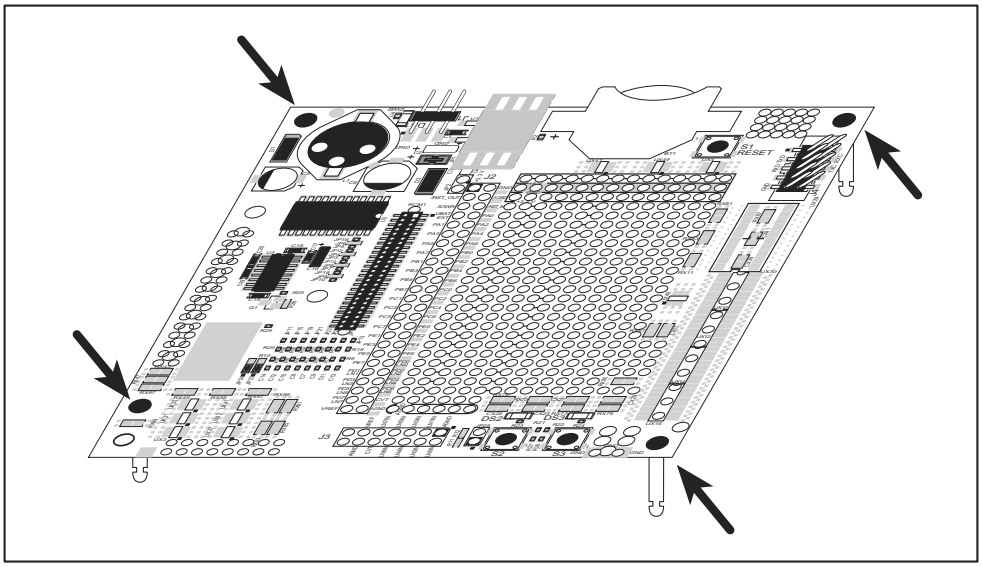

Figure 2. Insert Standoffs

2.3 Run a Sample Program

Once the RCM4000/RCM4010/RCM4050 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 RCM4000/RCM4010/RCM4050, 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 dropdown 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. 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 in the Dynamic C **Options > Project Options > Communications** menu. Choose a lower debug baud rate. Click **OK** to save.

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 RCM4000 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 connector 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 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 RCM4000 series of modules and the Prototyping Board.

For advanced development topics, refer to the ***Dynamic C User's Manual***, also in the online documentation set.

2.4.1 Technical Support

NOTE: If you purchased your RCM4000/RCM4010/RCM4050 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/.

3.2.3 A/D Converter Inputs (RCM4000 only)

The following sample programs are found in the **SAMPLES\RCM4000\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 rewritten into the user block data area.

  Connect a positive voltage (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 should not be used with this sample program.

  NOTE: The above sample program will overwrite any 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 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. Remember that analog input LN7 on the Prototyping Board is used with the thermistor and should not be used with this sample program.

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

- **AD_RDVOLT_ALL.C**—Demonstrates how to read all single-ended A/D input channels using previously defined calibration constants. Coefficients are read from the simulated EEPROM in the flash memory to compute equivalent voltages, and cannot be run in RAM.

  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. Computed raw data and equivalent voltages will be displayed.

- **AD_SAMPLE.C**—Demonstrates how to 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 the A/D converter channels (except LN7). Coefficients are read from the simulated EEPROM in the flash memory to compute equivalent voltages, so the sample program cannot be run in RAM.

  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. Computed raw data and 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.

Figure 7 shows the use of the Rabbit 4000 microprocessor ports in the RCM4000 modules.

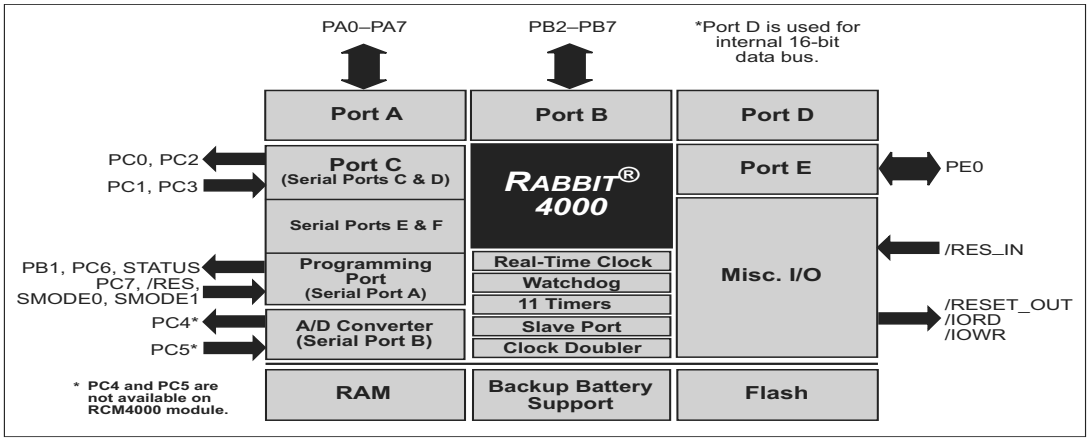

Figure 7. Use of Rabbit 4000 Ports

The ports on the Rabbit 4000 microprocessor used in the RCM4000 are configurable, and so the factory defaults can be reconfigured. Table 2 lists the Rabbit 4000 factory defaults and the alternate configurations.

4.3 Programming Cable

The programming cable is used to connect the programming port of the RCM4000 to a PC serial COM port. The programming cable converts the RS-232 voltage levels used by the PC serial port to the CMOS voltage levels used by the Rabbit 4000.

When the **PROG** connector on the programming cable is connected to the programming port on the RCM4000, programs can be downloaded and debugged over the serial interface.

The **DIAG** connector of the programming cable may be used on header J1 of the RCM4000 with the RCM4000 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 RCM4000 is automatically in Program Mode when the **PROG** connector on the programming cable is attached, and is automatically in Run Mode when no programming cable is attached. When the Rabbit 4000 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 4000 in the Program Mode. When the programming cable's **PROG** connector is not attached, the SMODE pins are pulled low, causing the Rabbit 4000 to operate in the Run Mode.

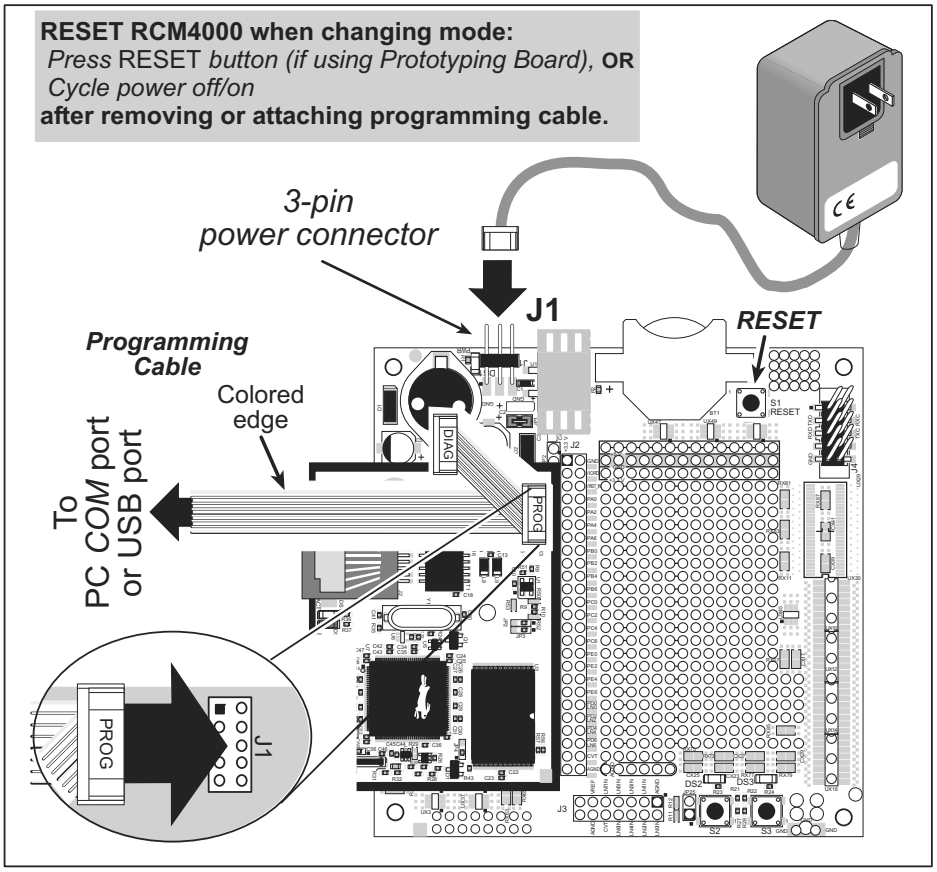

Figure 9. Switching Between Program Mode and Run Mode

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.

PARAMETERS

instructionbyte the instruction byte that will initiate a read or write operation at 8 or 16 bits on the designated register address. For example,

```
checkid = anaInConfig(0x5F, 0, 9600);
   // read ID and set baud rate
```

cmd the command data that configure the registers addressed by the instruction byte. Enter 0 if you are performing a read operation. For example,

```
i = anaInConfig(0x07, 0x3b, 0);
   // write ref/osc reg and enable
```

brate the serial clock transfer rate of 9600 to 115,200 bytes per second. **brate** must be set the first time this function is called. Enter 0 for this parameter thereafter, for example,

```
anaInConfig(0x00, 0x00, 9600);
   // resets device and sets byte rate
```

RETURN VALUE

0 on write operations

data value on read operations

SEE ALSO

anaInDriver, **anaIn**, **brdInit**

anaInCalib

```
int anaInCalib(int channel, int opmode, int gaincode,
   int value1, float volts1, int value2, float volts2);
```

DESCRIPTION

Calibrates the response of the desired A/D converter channel as a linear function using the two conversion points provided. Four values are calculated and placed into global tables **_adcCalibS**, **_adcCalibD**, and **adcCalibM** to be later stored into simulated EEPROM using the function **anaInEEWr()**. Each channel will have a linear constant and a voltage offset.

PARAMETERS

channel the channel number (0 to 7) corresponding to LN0 to LN7.

opmode the mode of operation:

SINGLE—single-ended input
DIFF—differential input
mAMP—4–20 mA input

channel	SINGLE	DIFF	mAMP
0	+AIN0	+AIN0 -AIN1	+AIN0*
1	+AIN1	+AIN1 -AIN0*	+AIN1*
2	+AIN2	+AIN2 -AIN3	+AIN2*
3	+AIN3	+AIN3 -AIN2*	+AIN3
4	+AIN4	+AIN4 -AIN5	+AIN4
5	+AIN5	+AIN5 -AIN4*	+AIN5
6	+AIN6	+AIN6 -AIN7*	+AIN6
7	+AIN7	+AIN7 -AIN6*	+AIN7*

* Not accessible on Prototyping Board.

gaincode the gain code of 0 to 7 (applies only to Prototyping Board):

Gain Code	Gain Multiplier	Voltage Range (V)
0	×1	0–22.5
1	×2	0–11.25
2	×4	0–5.6
3	×5	0–4.5
4	×8	0–2.8
5	×10	0–2.25
6	×16	0–1.41
7	×20	0–1.126

value1 the first A/D converter channel raw count value (0–2047)

volts1 the voltage or current corresponding to the first A/D converter channel value (0 to +20 V or 4 to 20 mA)

value2 the second A/D converter channel raw count value (0–2047)

volts2 the voltage or current corresponding to the first A/D converter channel value (0 to +20 V or 4 to 20 mA)

RETURN VALUE

0 if successful.

RETURN VALUE

0 if successful.

-1 if not able to make calibration constants.

SEE ALSO

anaIn, **anaInVolts**, **anaInmAmps**, **anaInDiff**, **anaInCalib**, **brdInit**

anaInmAmps

```
float anaInmAmps(unsigned int channel);
```

DESCRIPTION

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

PARAMETERS

`channel` the channel number (0 to 7) corresponding to LN0 to LN7.

Channel Code	4–20 mA Input Lines*
0	+AIN0
1	+AIN1
2	+AIN2
3	+AIN3†
4	+AIN4*
5	+AIN5*
6	+AIN6*
7	+AIN7

* Negative input is ground.
† Applies to Prototyping Board.

RETURN VALUE

A current value between 4.00 and 20.00 mA corresponding to the current on the analog input channel.

`ADTIMEOUT` (-4095) if the conversion is incomplete or busy bit timeout.

`ADOVERFLOW` (-4096) for overflow or out of range.

SEE ALSO

`anaInCalib`, **`anaIn`**, **`anaInVolts`**

6.2.3 Dynamically Assigned Internet Addresses

In many instances, devices on a network do not have fixed IP addresses. This is the case when, for example, you are assigned an IP address dynamically by your dial-up Internet service provider (ISP) or when you have a device that provides your IP addresses using the Dynamic Host Configuration Protocol (DHCP). The RCM4000 modules can use such IP addresses to send and receive packets on the Internet, but you must take into account that this IP address may only be valid for the duration of the call or for a period of time, and could be a private IP address that is not directly accessible to others on the Internet. These addresses can be used to perform some Internet tasks such as sending e-mail or browsing the Web, but it is more difficult to participate in conversations that originate elsewhere on the Internet. If you want to find out this dynamically assigned IP address, under Windows 98 you can run the **winipcfg** program while you are connected and look at the interface used to connect to the Internet.

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

If you are not concerned about accessing the RCM4000 from the Internet, you can place the RCM4000 on the internal network using an IP address assigned either statically or through DHCP.

6.3 Placing Your Device on the Network

In many corporate settings, users are isolated from the Internet by a firewall and/or a proxy server. These devices attempt to secure the company from unauthorized network traffic, and usually work by disallowing traffic that did not originate from inside the network. If you want users on the Internet to communicate with your RCM4000, you have several options. You can either place the RCM4000 directly on the Internet with a real Internet address or place it behind the firewall. If you place the RCM4000 behind the firewall, you need to configure the firewall to translate and forward packets from the Internet to the RCM4000.

A.1.1 A/D Converter

Table A-2 shows some of the important A/D converter specifications. For more details, refer to the ADC7870 data sheet.

Table A-2. A/D Converter Specifications

Parameter	Test Conditions	Typ	Max
Analog Input Characteristics			
Input Capacitance		4 – 9.7 pF	
Input Impedance			
Common-Mode		6 MΩ	
Differential Mode		7 MΩ	
Static Accuracy			
Resolution			
Single-Ended Mode		11 bits	
Differential Mode		12 bits	
Integral Linearity		±1 LSB	±2.5 LSB
Differential Linearity		±0.5 LSB	
Dynamic Characteristics			
Throughput Rate		52 ksamples/s	
Voltage Reference			
Accuracy	V_{ref} = 2.048 V and 2.5 V	±0.05%	±0.25%
Buffer Amp Source Current		20 mA	
Buffer Amp Sink Current		200 µA	
Short-Circuit Current		20 mA	

A.3 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 29.4 MHz CPU clock with the clock doubler enabled and capacitive loading on address and data lines of less than 70 pF per pin. The absolute maximum operating voltage on all I/O is 3.6 V.

A.4 Bus Loading

You must pay careful attention to bus loading when designing an interface to the RCM4000. This section provides bus loading information for external devices.

Table A-5 lists the capacitance for the various RCM4000 I/O ports.

Table A-5. Capacitance of Rabbit 4000 I/O Ports

I/O Ports	Input Capacitance (pF)	Output Capacitance (pF)
Parallel Ports A to E	12	14

Table A-6 lists the external capacitive bus loading for the various RCM4000 output ports. Be sure to add the loads for the devices you are using in your custom system and verify that they do not exceed the values in Table A-6.

Table A-6. External Capacitive Bus Loading -40°C to +85°C

Output Port	Clock Speed (MHz)	Maximum External Capacitive Loading (pF)
All I/O lines with clock doubler enabled	58.98	100

Table A-7 lists the loadings for the A/D converter inputs.

Table A-7. A/D Converter Inputs

Parameter	Value
Input Capacitance	4–9.7 pF
Input Impedance	Common-Mode 6 MΩ Differential 7 MΩ

SCHEMATICS

090-0227 RCM4000 Schematic

www.rabbit.com/documentation/schemat/090-0227.pdf

090-0230 Prototyping Board Schematic

www.rabbit.com/documentation/schemat/090-0230.pdf

090-0128 Programming Cable Schematic

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

090-0252 USB Programming Cable Schematic

www.rabbit.com/documentation/schemat/090-0252.pdf

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