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 2000
Co-Processor	-
Speed	25.8MHz
Flash Size	256KB
RAM Size	128KB
Connector Type	2 IDC Headers 2x20
Size / Dimension	1.9" x 2.3" (48.3mm x 58.4mm)
Operating Temperature	-40°C ~ 85°C
Purchase URL	https://www.e-xfl.com/product-detail/digi-international/20-101-0405

- External reset input
- Reset output
- Five 8-bit timers, two 10-bit timers; five timers are cascadable in pairs
- 256K flash EPROM, 512K SRAM
- Real-time clock
- Watchdog supervisor
- Provision for customer-supplied backup battery via connections on header J2
- Four CMOS-compatible serial ports: maximum asynchronous baud rate of 806,400 bps, maximum synchronous baud rate of 6.45 Mbps. Two ports are configurable as clocked ports.

Appendix A, “Specifications,” provides detailed specifications for the RCM2000.

Three versions of the RCM2000 are available. Their standard features are summarized in Table 1.

Table 1. RCM2000 Models and Features

Model	Features
RCM2000	Full-featured RCM2000 module with 25.8 MHz clock, 256K flash memory, and 512K SRAM
RCM2010	RCM2000 with 25.8 MHz clock and 128K SRAM
RCM2020	RCM2000 with 18.432 MHz clock and 128K SRAM

1.2 Advantages of Using the RCM2000

- Fast design time for your project since the basic core has already been designed and built.
- Competitive pricing compared with purchasing and assembling the individual components.
- Easy programming, including production installation of a program.
- Generous memory size allows large C programs with tens of thousands of lines of code, and substantial data storage.

2. HARDWARE SETUP

This chapter describes the RCM2000 hardware in more detail, and explains how to set up the accompanying Prototyping Board.

NOTE: This chapter (and this manual) assume that you have the RabbitCore RCM2000 Development Kit. If you purchased an RCM2000 module by itself, you will have to adapt the information in this chapter and elsewhere to your test and development setup.

2. Connect RCM2000 to PC

Connect the 10-pin connector of the programming cable labeled **PROG** to header J3 on the RCM2000 module as shown in Figure 2 below. Be sure to orient the 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.)

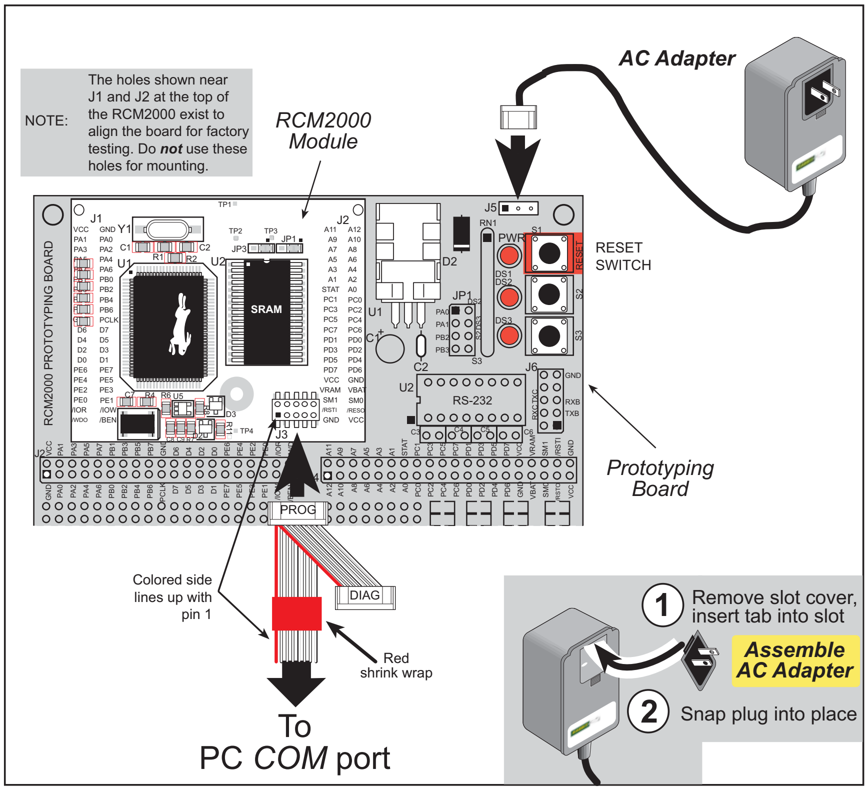


Figure 2. RCM2000 Power and Programming Connections

NOTE: Some PCs now come equipped only with a USB port. It may be possible to use an RS-232/USB converter (Part No. 20-151-0178) with the programming cable supplied with the RCM2000 Development Kit. Note that not all RS-232/USB converters work with Dynamic C.

3. Power Supply Connections

When all 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 RCM2000 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 2, then press down on the spring-loaded clip below the plug assembly to allow the plug assembly to click into place.

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

Plug in the AC adapter. The power LED on the Prototyping Board should light up. The RCM2000 and the Prototyping Board are now ready to be used.

> **NOTE:** A RESET button is provided on the Prototyping Board to allow a hardware reset.

To power down the Prototyping Board, unplug the power connector from J5. You should disconnect power before making any circuit adjustments in the prototyping area, changing any connections to the board, or removing the RCM2020 from the Prototyping Board.

2.1.1 Alternate Power Supply Connections

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

3.1.1 Running Sample Program FLASHLED.C

This sample program will be used to illustrate some of the functions of Dynamic C.

First, open the file **`FLASHLED.C`**, which is in the **`SAMPLES/RCM2000`** folder. The program will appear in a window, as shown in Figure 3 below (minus some comments). Use the mouse to place the cursor on the function name **`WrPortI`** in the program and type **<Ctrl-H>**. This will bring up a documentation box for the function **`WrPortI`**. In general, you can do this with all functions in Dynamic C libraries, including libraries you write yourself. Close the documentation box and continue.

```
main(){

  int j;

  WrPortI(SPCR,&SPCRShadow,0x84);
  WrPortI(PADR,&PADRShadow,0xFF);

    while(1) {

      BitWrPortI(PADR,&PADRShadow,1,1);

      for(j=0; j<32000; j++);

      BitWrPortI(PADR,&PADRShadow,0,1);

      for(j=0; j<25000; j++);

    } // end while

 } //  end of main
```

C programs begin with main

Set up Port A to output to LED DS2 and DS3

Start a loop

Turn LED DS3 off

Time delay by counting to 32,000

Turn LED DS3 on

Time delay by counting to 25,000

End of the endless loop

Note: See the ***Rabbit 2000 Microprocessor User's Manual*** (Software Chapter) for details on the routines that read and write I/O ports.

Figure 3. Sample Program FLASHLED.C

To run the program **`FLASHLED.C`**, open it with the **File** menu (if it is not already open), then compile and run it by pressing **F9** or by selecting **Run** in the **Run** menu. The LED on the Prototyping Board should start flashing if everything went well. If this doesn't work review the following points.

- The target should be ready, which is indicated by the message "BIOS successfully compiled..." If you did not receive this message or you get a communication error, recompile the BIOS by typing **<Ctrl-Y>** or select **Recompile BIOS** from the **Compile** menu.

- Setting break points. The **F2** key is used to turn on or turn off (toggle) a break point at the cursor position if the program has already been compiled. You can set a break point if the program is paused at a break point. You can also set a break point in a program that is running at full speed. This will cause the program to break if the execution thread hits your break point.
- Watch expressions. A watch expression is a C expression that is evaluated on command in the watch window. An expression is basically any type of C formula that can include operators, variables and function calls, but not statements that require multiple lines such as *for* or *switch*. You can have a list of watch expressions in the watch window. If you are single-stepping, then they are all evaluated on each step. You can also command the watch expression to be evaluated by using the **<Ctrl-U>** command. When a watch expression is evaluated at a break point, it is evaluated as if the statement was at the beginning of the function where you are single-stepping. If your program is running you can also evaluate watch expressions with a **<Ctrl-U>** if your program has a **`runwatch()`** command that is frequently executed. In this case, only expressions involving global variables can be evaluated, and the expression is evaluated as if it were in a separate function with no local variables.

3.1.1.7 Cooperative Multitasking

Cooperative multitasking is a convenient way to perform several different tasks at the same time. An example would be to step a machine through a sequence of steps and at the same time independently carry on a dialog with the operator via a human interface. Cooperative multitasking differs from another approach called preemptive multitasking. Dynamic C supports both types of multitasking. In cooperative multitasking each separate task voluntarily surrenders its compute time when it does not need to perform any more activity immediately. In preemptive multitasking control is forcibly removed from the task via an interrupt.

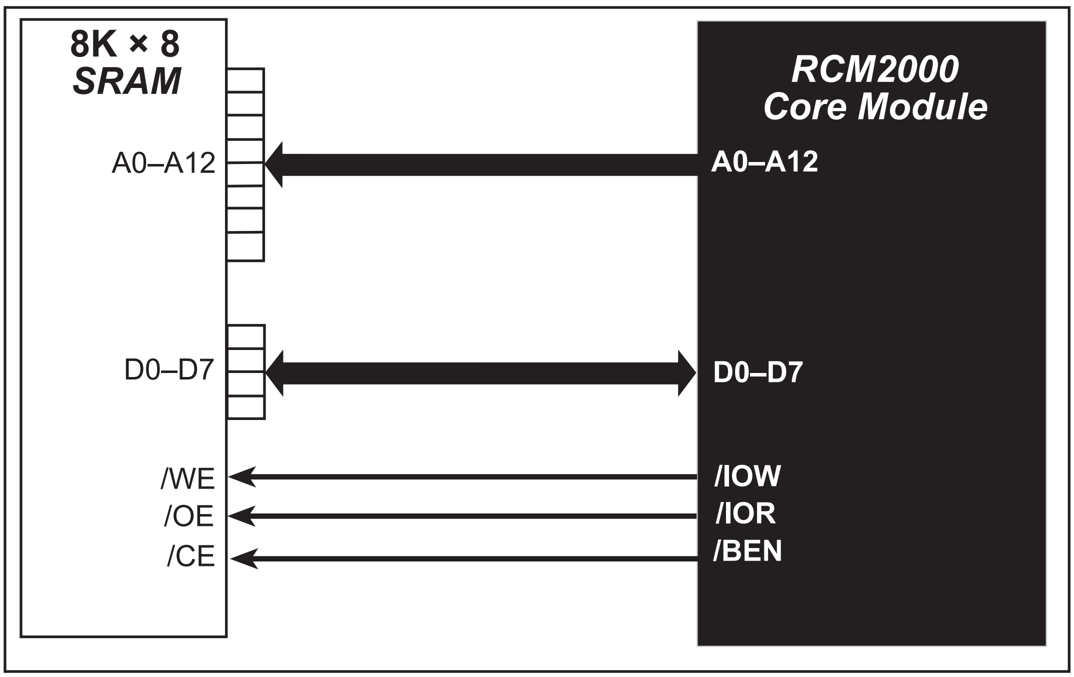
Dynamic C has language extensions to support multitasking. The major C constructs are called *costatements*, *cofunctions*, and *slicing*. These are described more completely in the ***Dynamic C User's Manual***. The example below, sample program **`FLASHLEDS2.C`**, uses costatements. A costatement is a way to perform a sequence of operations that involve pauses or waits for some external event to take place. A complete description of costatements is in the ***Dynamic C User's Manual***. The **`FLASHLEDS2.C`** sample program has two independent tasks. The first task flashes LED DS2 2.5 times a second. The second task flashes DS3 every 1.5 seconds.

3.1.2 Getting to Know the RCM2000

The following sample programs can be found in the **`SAMPLES\RCM2000`** folder.

- **`EXTSRAM.C`**—demonstrates the setup and simple addressing to an external SRAM. This program first maps the external SRAM to the I/O Bank 0 register with a maximum of 15 wait states, chip select strobe (which is ignored because of the circuitry), and allows writes. The first 256 bytes of SRAM are cleared and read back. Values are then written to the same area and are read back. The Dynamic C **STDIO** window will indicate if writes and reads did not occur

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



- **`FLASHLED.C`**—repeatedly flashes LED DS3 on the Prototyping Board on and off. LED DS3 is controlled by Parallel Port A bit 1 (PA1).
- **`FLASHLED2.C`**—repeatedly flashes LED DS3 on the Prototyping Board on and off. LED DS3 is controlled by Parallel Port A bit 1 (PA1).

  This sample program also shows the use of the **`runwatch()`** function to allow Dynamic C to update watch expressions while running. The following steps explain how to do this.

  1. Add a watch expression for "k" in the **Inspect > Add Watch** dialog box.
  2. Click "Add" or "Add to top" so that it will be in the watch list permanently.
  3. Click **OK** to close the dialog box.
  4. Press **<Ctrl+U>** while the program is running. This will update the watch window

4. HARDWARE REFERENCE

Chapter 4 describes the principal subsystems for the RCM2000.

4.1 RCM2000 Digital Inputs and Outputs

Figure 4 shows the subsystems designed into the RCM2000.

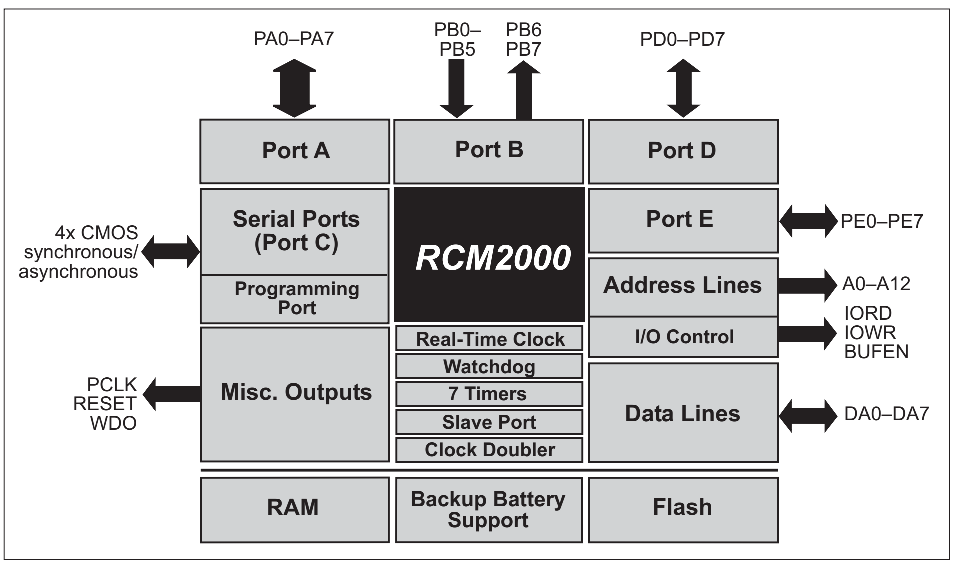


Figure 4. Rabbit Subsystems

4.3.2 Programming Port

The RCM2000 has a 10-pin program header labeled J3. The programming port uses the Rabbit 2000's Serial Port A for communication. Dynamic C uses the programming port to download and debug programs.

The programming port is also used for the following operations.

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

Alternate Uses of the Serial Programming Port

All three clocked Serial Port A signals are available as

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

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

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

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

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

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

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

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

4.5 Other Hardware

4.5.1 Clock Doubler

The RCM2000 takes advantage of the Rabbit 2000 microprocessor's internal clock doubler. A built-in clock doubler allows half-frequency crystals to be used to reduce radiated emissions. The 25.8 MHz (RCM 2000 and RCM2010) and 18.4 MHz (RCM 2020) frequencies are generated using 12.9 MHz and 9.2 MHz crystals. The clock doubler is disabled automatically in the BIOS for crystals with a frequency above 12.9 MHz.

The clock doubler can be disabled if 25.8 MHz or 18.4 MHz clock speeds are not required. Disabling the Rabbit 2000 microprocessor's internal clock will reduce power consumption and further reduce radiated emissions. The clock doubler is disabled with a simple configuration macro as shown below.

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

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

Change the serial baud rate to 57,600 bps when the RCM2000 is operated at 12.9 MHz or 9.2 MHz.

4.6 Memory

4.6.1 SRAM

The RCM2000 is designed to accept 32K to 512K of SRAM packaged in an SOIC case.

4.6.2 Flash EPROM

The RCM2000 is also designed to accept 128K to 512K of flash EPROM packaged in a TSOP case.

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

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

A Flash Memory Bank Select jumper configuration option based on 0 Ω surface-mounted resistors exists at header JP3. This option, used in conjunction with some configuration macros, allows Dynamic C to compile two different co-resident programs for the upper and lower halves of the 512K flash in such a way that both programs start at logical address 0000. This is useful for applications that require a resident download manager and a separate downloaded program. See Technical Note 218, ***Implementing a Serial Download Manager for a 256K Flash***, for details.

4.6.3 Dynamic C BIOS Source Files

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

5.1.1 Using Dynamic C

You have a choice of doing your software development in the flash memory or in the SRAM included on the RCM2000. The flash memory and SRAM options are selected with the **Options > Project Options > Compiler** menu.

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

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

> **NOTE:** Do not depend on the flash memory sector size or type. Due to the volatility of the flash memory market, the RCM2000 and Dynamic C were designed to accommodate flash devices with various sector sizes.

Developing software with Dynamic C is simple. Users can write, compile, and test C and assembly code without leaving the Dynamic C development environment. Debugging occurs while the application runs on the target. Alternatively, users can compile a program to an image file for later loading. Dynamic C runs on PCs under Windows 95, 98, 2000, NT, Me, and XP. Programs can be downloaded at baud rates of up to 460,800 bps after the program compiles.

Dynamic C has a number of standard features.

- Full-feature source and/or assembly-level debugger, no in-circuit emulator required.
- Royalty-free TCP/IP stack with source code and most common protocols.
- Hundreds of functions in source-code libraries and sample programs:
  - ▶ Exceptionally fast support for floating-point arithmetic and transcendental functions.
  - ▶ RS-232 and RS-485 serial communication.
  - ▶ Analog and digital I/O drivers.
  - ▶ I^2C, SPI, GPS, file system.
  - ▶ LCD display and keypad drivers.
- Powerful language extensions for cooperative or preemptive multitasking.
- Loader utility program to load binary images into Rabbit targets in the absence of Dynamic C.
- Provision for customers to create their own source code libraries and augment on-line help by creating “function description” block comments using a special format for library functions.

A.1 Electrical and Mechanical Specifications

Figure A-1 shows the mechanical dimensions for the RCM2000.

Please refer to the RCM2000 footprint diagram later in this appendix for precise header locations.

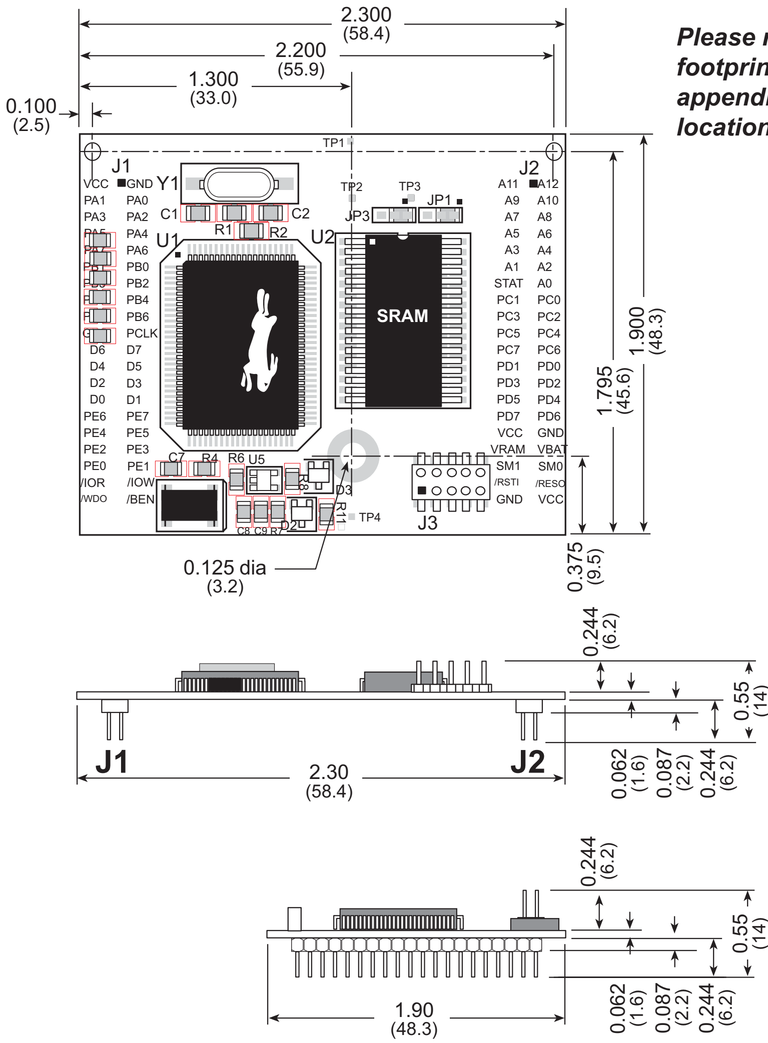


Figure A-1. RCM2000 Dimensions

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

It is recommended that you allow for an “exclusion zone” of 0.04" (1 mm) around the RCM2000 in all directions when the RCM2000 is incorporated into an assembly that includes other printed circuit boards. An “exclusion zone” of 0.08" (2 mm) is recommended below the RCM2000 when the RCM2000 is plugged into another assembly using the shortest connectors for header J1. Figure A-2 shows this “exclusion zone.”

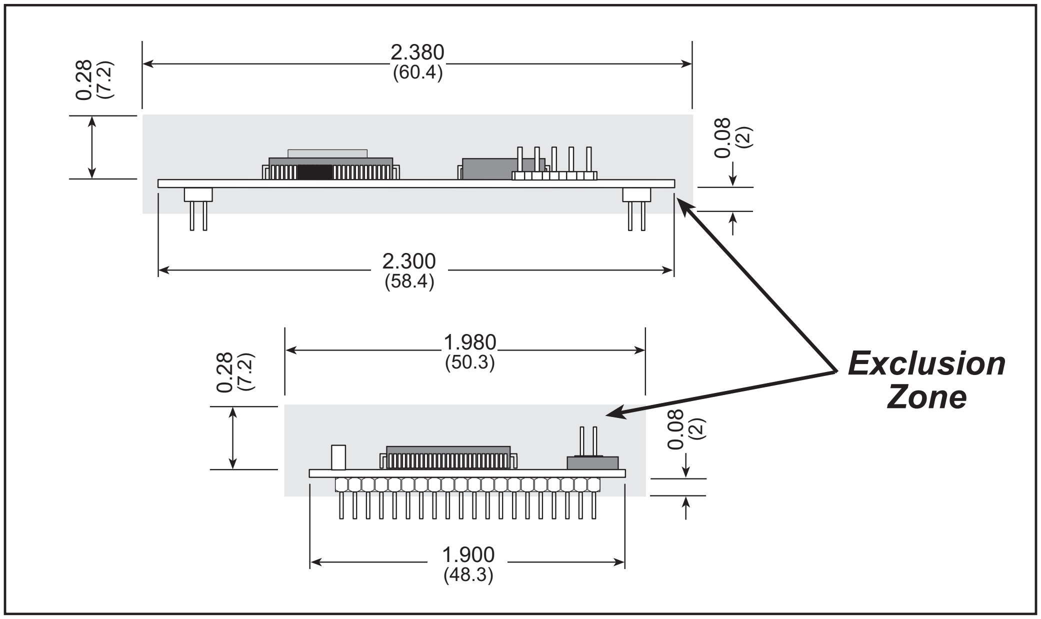


Figure A-2. RCM2000 “Exclusion Zone”

A.2 Bus Loading

You must pay careful attention to bus loading when designing an interface to the RCM2000. This section provides bus loading for external devices.

Table A-2 lists the capacitance for the various RCM2000 I/O ports.

Table A-2. Capacitance of RCM2000 I/O Ports

I/O Ports	Input Capacitance (pF)		Output Capacitance (pF)	
	Typ.	Max.	Typ.	Max.
Parallel Ports A to E	6 pF	12 pF	10 pF	14 pF
Data Lines D0–D7	16 pF	30 pF	24 pF	32 pF
Address Lines A0–A12	—	—	24 pF	32 pF

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

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

Output Port	Clock Speed (MHz)	Maximum External Capacitive Loading (pF)
A[12:1] D[7:1]	25.8	50
A[12:1] D[7:1]	18.4	55 for 90 ns flash 100 for 55 ns flash*
A0 D0	25.8, 18.4	100
PD[3:0]	25.8, 18.4,	100
PA[7:0] PB[7,6] PC[6,4,2,0] PD[7:4] PE[7:0]	25.8, 18.4	90
All data, address, and I/O lines with clock doubler disabled	12.9, 9.2	100

* The RCM2020 operating at 18.4 MHz will typically come with a flash EPROM whose access time is 55 ns. Because of the volatility of the memory market, a 90 ns flash EPROM could be used on the RCM2020.

B.4 Using the Prototyping Board

The Prototyping Board is actually both a demonstration board and a prototyping board. As a demonstration board, it can be used to demonstrate the functionality of the RCM2000 right out of the box without any modifications to either board. There are no jumpers or dip switches to configure or misconfigure on the Prototyping Board so that the initial setup is very straightforward.

The Prototyping Board comes with the basic components necessary to demonstrate the operation of the RCM2000. Two LEDs (DS2 and DS3) are connected to PA0 and PA1, and two switches (S2 and S3) are connected to PB2 and PB3 to demonstrate the interface to the Rabbit 2000 microprocessor. Reset switch S1 is the hardware reset for the RCM2000.

To maximize the availability of RCM2000 resources, the demonstration hardware (LEDs and switches) on the Prototyping Board may be disconnected. This is done by cutting the traces below the silk-screen outline of header JP1 on the bottom side of the Prototyping Board. Figure B-4 shows the four places where cuts should be made. An exacto knife would work nicely to cut the traces. Alternatively, a small standard screwdriver may be carefully and forcefully used to wipe through the PCB traces.

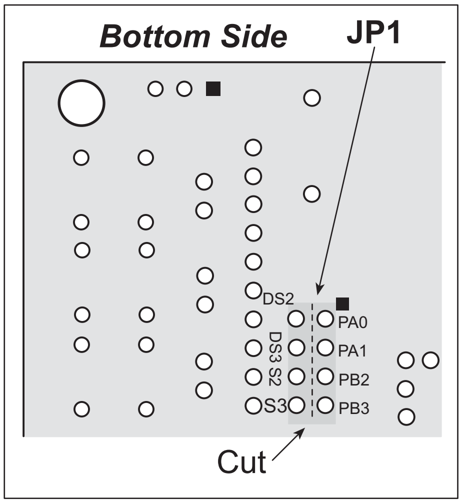


Figure B-4. Where to Cut Traces to Permanently Disable Demonstration Hardware on Prototyping Board

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

D.5 Simple D/A Converter

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

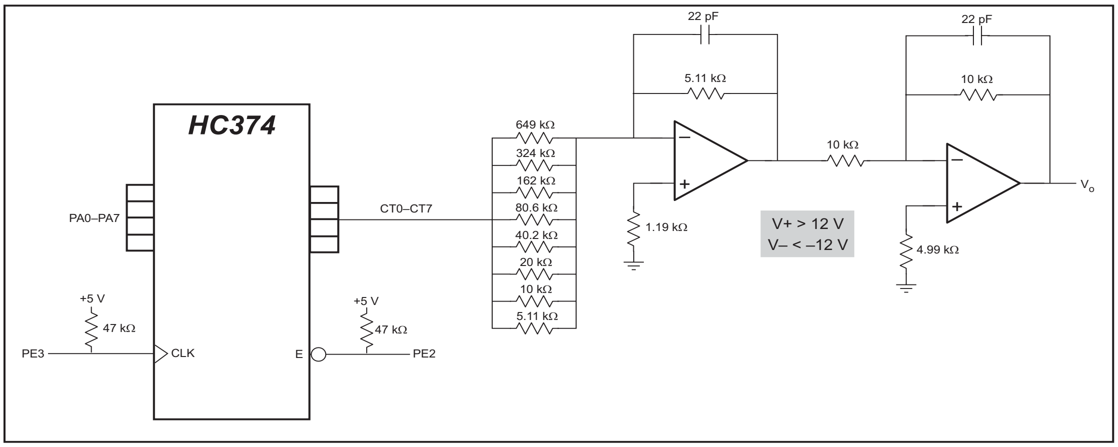


Figure D-6. Sample D/A Converter Connections

A sample program is not available at this time.

INDEX

SCHEMATICS

090-0097 RCM2000 Schematic

www.rabbit.com/documentation/schemat/090-0097.pdf

090-0099 RCM2000 Prototyping Board Schematic

www.rabbit.com/documentation/schemat/090-0099.pdf

090-0128 Programming Cable Schematic

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

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