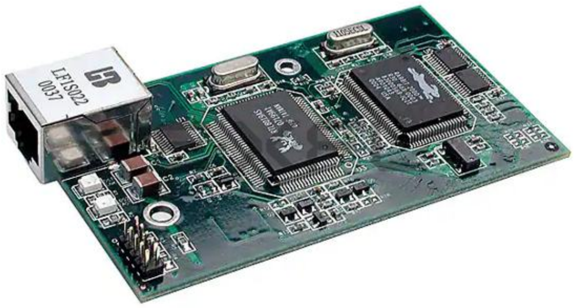

Welcome to E-XFL.COM

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	22.1MHz
Flash Size	512KB
RAM Size	512KB
Connector Type	2 IDC Headers 2x20
Size / Dimension	2" x 3.5" (51mm x 89mm)
Operating Temperature	-40°C ~ 70°C
Purchase URL	https://www.e-xfl.com/product-detail/digi-international/101-0434

1.2 Advantages of the RCM2100

- Fast time to market using a fully engineered, “ready to run” microprocessor core.
- Competitive pricing when compared with the alternative of purchasing and assembling individual components.
- Easy C-language program development and debugging, including rapid production loading of programs.
- Generous memory size allows large programs with tens of thousands of lines of code, and substantial data storage.
- Integrated Ethernet port (on selected models) for network connectivity, royalty-free TCP/IP software.
- Models with and without Ethernet for flexible production options.
- Small size and identical footprint and pinout for all models.

3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM2100 (and for all other Rabbit Semiconductor hardware), you must install and use Dynamic C. Dynamic C is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Rabbit Semiconductor single-board computers and other single-board computers based on the Rabbit microprocessor. This chapter takes you through the installation of Dynamic C, and then provides a tour of the sample programs for the RCM2100.

3.1 Sample Programs

To help familiarize you with the RCM2100 modules, several sample Dynamic C programs have been included. Loading, executing and studying these programs will give you a solid hands-on overview of the RCM2100's capabilities, as well as a quick start with Dynamic C as an application development tool. These programs are intended to serve as tutorials, but then can also be used as starting points or building blocks for your own applications.

> **NOTE:** It is assumed in this section that you have at least an elementary grasp of ANSI C. If you do not, see the introductory pages of the ***Dynamic C User's Manual*** for a suggested reading list.

Each sample program has comments that describe the purpose and function of the program.

Before running any of these sample programs, make sure that your RCM2100 is connected to the Prototyping Board and to your PC as described in Section 2.1, "Connections." To run a sample program, open it with the **File** menu (if it is not already open), then compile and run it by pressing **F9** or by selecting **Run** in the **Run** menu.

Sample programs are provided in the Dynamic C `SAMPLES` folder. Two folders contain sample programs that illustrate features unique to the RabbitCore RCM2100.

- `RCM2100`—Demonstrates the basic operation and the Ethernet functionality of the RabbitCore RCM2100.
- `TCPIP`—Demonstrates more advanced TCP/IP programming for Rabbit Semiconductor's Ethernet-enabled Rabbit-based boards.

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

The RCM2100 has 40 parallel I/O lines grouped in five 8-bit ports available on headers J1 and J2. The 24 bidirectional I/O lines are located on pins PA0–PA7, PD0–PD7, and PE0–PE7. The pinouts for headers J1 and J2 are shown in Figure 6.

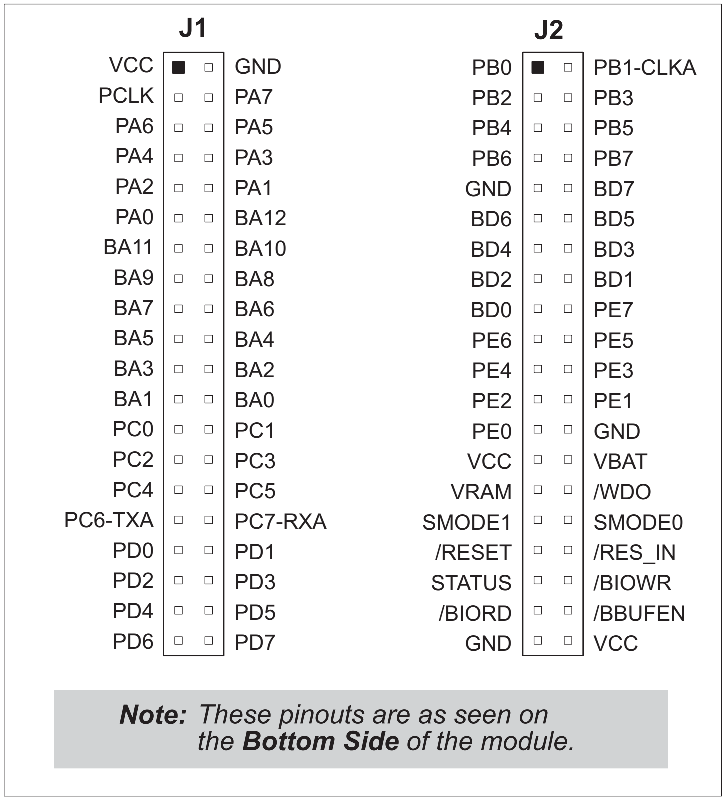

Figure 6. RCM2100 I/O Pinouts

The ports on the Rabbit 2000 microprocessor used in the RCM2100 are configurable, and so the factory defaults can be reconfigured. Table 2 lists the Rabbit 2000 factory defaults and the alternate configurations.

As shown in Table 2, pins PA0–PA7 can be used to allow the Rabbit 2000 to be a slave to another processor. PE0, PE1, PE4, and PE5 can be used as external interrupts INT0A, INT1A, INT0B, and INT1B. Pins PB0 and PB1 can be used to access the clock on Serial Port B and Serial Port A of the Rabbit microprocessor. Pins PD4 and PD6 can be programmed to be optional serial outputs for Serial Ports B and A. PD5 and PD7 can be used as alternate serial inputs by Serial Ports B and A.

The Ethernet-enabled versions of the RCM2100 do not have 0 Ω resistors (jumpers) installed at R21, R24, and R35–R38, which allows PE6, PE2, and PD4–PD7 to connect to the RealTek Ethernet chip that is stuffed on those versions.

Table 2. RCM2100 Pinout Configurations

<table>
<tr><th colspan="2">Pin</th><th>Pin Name</th><th>Default Use</th><th>Alternate Use</th><th>Notes</th></tr>
<tr><td rowspan="20">Header J1</td><td>1</td><td>VCC</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>GND</td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>PCLK</td><td>Output (Internal Clock)</td><td>Output</td><td>Turned off in software</td></tr>
<tr><td>4–11</td><td>PA[7:0]</td><td>Parallel I/O</td><td>Slave port data bus SD0–SD7</td><td></td></tr>
<tr><td>12–24</td><td>BA[12:0]</td><td>Output</td><td></td><td>Buffered Rabbit 2000 address bus</td></tr>
<tr><td>25</td><td>PC0</td><td>Output</td><td>TXD</td><td></td></tr>
<tr><td>26</td><td>PC1</td><td>Input</td><td>RXD</td><td></td></tr>
<tr><td>27</td><td>PC2</td><td>Output</td><td>TXC</td><td></td></tr>
<tr><td>28</td><td>PC3</td><td>Input</td><td>RXC</td><td></td></tr>
<tr><td>29</td><td>PC4</td><td>Output</td><td>TXB</td><td></td></tr>
<tr><td>30</td><td>PC5</td><td>Input</td><td>RXB</td><td></td></tr>
<tr><td>31</td><td>PC6</td><td>Output</td><td>TXA</td><td rowspan="2">Connected to programming port</td></tr>
<tr><td>32</td><td>PC7</td><td>Input</td><td>RXA</td></tr>
<tr><td>33–36</td><td>PD[0:3]</td><td rowspan="5">Bitwise or parallel programmable I/O, can be driven or open-drain output</td><td></td><td>16 mA sourcing and sinking current at full AC switching speed</td></tr>
<tr><td>37</td><td>PD4</td><td>ATXB output</td><td>Ethernet chip RSTDRV</td></tr>
<tr><td>38</td><td>PD5</td><td>ARXB input</td><td>Ethernet chip BD5</td></tr>
<tr><td>39</td><td>PD6</td><td>ATXA output</td><td>Ethernet chip BD6</td></tr>
<tr><td>40</td><td>PD7</td><td>ARXA input</td><td>Ethernet chip BD7</td></tr>
</table>

Table 2. RCM2100 Pinout Configurations (continued)

<table>
<tr><th colspan="2">Pin</th><th>Pin Name</th><th>Default Use</th><th>Alternate Use</th><th>Notes</th></tr>
<tr><td rowspan="8">Header J2</td><td rowspan="2">31–32</td><td rowspan="2">SMODE1, SMODE0</td><td>(0,0)—start executing at address zero</td><td></td><td>No programming cable attached</td></tr>
<tr><td>SMODE0 =1, SMODE1 = 1
Cold boot from asynchronous serial port A at 2400 bps (programming cable connected)</td><td>(0,1)—cold boot from slave port
(1,0)—cold boot from clocked serial port A</td><td>With programming cable attached</td></tr>
<tr><td>33</td><td>/RESET</td><td>Reset output</td><td></td><td></td></tr>
<tr><td>34</td><td>/RES_IN</td><td>Reset input</td><td></td><td></td></tr>
<tr><td>35</td><td>STATUS</td><td>Output (Status)</td><td>Output</td><td></td></tr>
<tr><td>36</td><td>/BIOWR</td><td>Output (I/O buffer write strobe)</td><td></td><td></td></tr>
<tr><td>37</td><td>/BIORD</td><td>Output (I/O buffered strobe)</td><td></td><td></td></tr>
<tr><td>38</td><td>/BUFEN</td><td>Output (I/O buffer enable)</td><td></td><td></td></tr>
</table>

4.3.2 Standalone Operation of the RCM2100

The RCM2100 must be programmed via the RCM2100 Prototyping Board or via a similar arrangement on a customer-supplied board. Once the RCM2100 has been programmed successfully, remove the programming cable from the programming connector and reset the RCM2100. The RCM2100 may be reset by cycling the power off/on or by pressing the **RESET** button on the Prototyping Board. The RCM2100 module may now be removed from the Prototyping Board for end-use installation.

> **CAUTION:** Power to the Prototyping Board or other boards should be disconnected when removing or installing your RCM2100 module to protect against inadvertent shorts across the pins or damage to the RCM2100 if the pins are not plugged in correctly. Do not reapply power until you have verified that the RCM2100 module is plugged in correctly.

4.4 Memory

4.4.1 SRAM

The RCM2100 is designed to accept 32K to 512K of SRAM packaged in an SOIC case.

4.4.2 Flash Memory

The RCM2100 is also designed to accept 128K to 512K of flash memory packaged in a TSOP case.

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

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

A Flash Memory Bank Select jumper configuration option based on 0 Ω surface-mounted resistors exists at header JP4. 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.

> **NOTE:** Only the Normal Mode (pins 1–2 connected at JP4), which corresponds to using the full code space, is supported at the present time.

4.4.3 Dynamic C BIOS Source Files

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

5. Software Reference

Dynamic C is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with Rabbit Semiconductor single-board computers and other single-board computers based on the Rabbit microprocessor. Chapter 4 provides the libraries and function calls related to the RCM2100.

5.1 More About Dynamic C

Dynamic C has been in use worldwide since 1989. Dynamic C is specially designed for programming embedded systems, and features quick compile and interactive debugging in the real environment. A complete reference to Dynamic C is contained in the ***Dynamic C User's Manual***.

Dynamic C for Rabbit® processors uses the standard Rabbit programming interface. This is a 10-pin connector that connects to the Rabbit Serial Port A. It is possible to reset and cold-boot a Rabbit processor via the programming port. No software needs to be present in the target system. More details are available in the ***Rabbit 2000 Microprocessor User's Manual***.

Dynamic C cold-boots the target system and compiles the BIOS. The BIOS is a basic program of a few thousand bytes in length that provides the debugging and communication facilities that Dynamic C needs. Once the BIOS has been compiled, the user can compile his own program and test it. If the BIOS fails because the program stops running, a new cold boot and BIOS compile can be done at any time.

The BIOS can be customized by using **`#define`** options.

Dynamic C does not use **`include`** files, rather it has libraries that are used for the same purpose, that is, to supply function prototypes to programs before they are compiled. See Section 4.24, “Modules,” in the ***Dynamic C User's Manual*** for more information.

Dynamic C supports assembly language, either as separate functions or as fragments embedded in C programs. Interrupt routines may be written in Dynamic C or in assembly language.

5.2 I/O

The RCM2100 was designed to interface with other systems, and so there are no drivers written specifically for the I/O. The general Dynamic C read and write functions allow you to customize the parallel I/O to meet your specific needs. For example, use

```
WrPortI(PEDDR, &PEDDRShadow, 0x00);
```

to set all the port E bits as inputs, or use

```
WrPortI(PEDDR, &PEDDRShadow, 0xFF);
```

to set all the port E bits as outputs.

The sample programs in the Dynamic C **SAMPLES/RCM2100** directory provide further examples.

5.2.1 PCLK Output

The PCLK output is controlled by bits 7 and 6 of the Global Output Register (GOCR) on the Rabbit 2000 microprocessor, and so can be enabled or disabled in software. Starting with Dynamic C v 7.02, the PCLK output is disabled by default at compile time to minimize radiated emissions; the PCLK output is enabled in earlier versions of Dynamic C.

Use the following code to set the PCLK output as needed.

PCLK output driven with peripheral clock:

```
WrPortI(GOCR, &GOCRShadow, (GOCRShadow&~0xc0));
```

PCLK output driven with peripheral clock ÷ 2:

```
WrPortI(GOCR, &GOCRShadow, ((GOCRShadow&~0xc0)| 0x40));
```

PCLK output off (low):

```
WrPortI(GOCR, &GOCRShadow, ((GOCRShadow&~0xc0)| 0x80));
```

PCLK output on (high):

```
WrPortI(GOCR, &GOCRShadow, (GOCRShadow | 0xc0));
```

6.10.2 Sample Program: ETHCORE1.C

The RCM2100 modules with Ethernet ports can act as micro Web page servers, with dynamic interaction between the controller and the web pages. This sample program demonstrates how a Web page can be used to both monitor and control an RCM2100 module.

Compile & Run Program

Open the sample program **`ETHCORE1.C`**. Press **F9** to compile and run the program.

> **TIP:** This program will be more interesting to observe if LEDs DS4 and DS5 are installed on the Prototyping Board.

When the program starts, LEDs DS2, DS3 and DS5 will be lit, and DS4 will be dark. Open a web browser and enter the IP address you defined for the RCM2100 module in the program in the address window. A page like that shown in Figure 11 should appear.

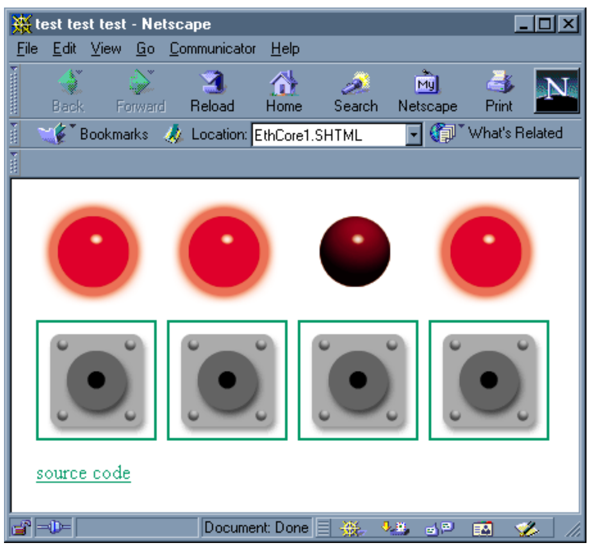

Figure 11. Browser screen for Sample Program **`ETHCORE1.C`**

Clicking on each of the button images in the browser window will toggle the state of the associated LED image, and will toggle the state of the corresponding LED on the Prototyping Board. Since the web page is generated by the RabbitCore module (using Dynamic HTML), the LED image and the corresponding LED's real state will always be in step.

Program Description

This program begins to show the range of applications for an Ethernet-enabled embedded system controller, so let's look closely at its operation.

As with **`PINGLED.C`**, several network addresses must be defined before this application can work. The **`TCPCONFIG 1`** macro in the sample program provides default settings for **`MY_IP_ADDRESS`**, which is the address of the RCM2100 module, **`MY_NETMASK`**, and **`MY_GATEWAY`** (which needs to be defined if you wish to reach the system from outside the local network). These TCP/IP settings can be changed as needed in the **`TCP_CONFIG.LIB`** library.

It is recommended that you allow for an “exclusion zone” of 0.04" (1 mm) around the RCM2100 in all directions when the RCM2100 is incorporated into an assembly that includes other components. An “exclusion zone” of 0.16" (4 mm) is recommended below the RCM2100 when the RCM2100 is plugged into another assembly using the shortest connectors for headers J1 and J2 on the RCM2100. Figure A-2 shows this “exclusion zone.”

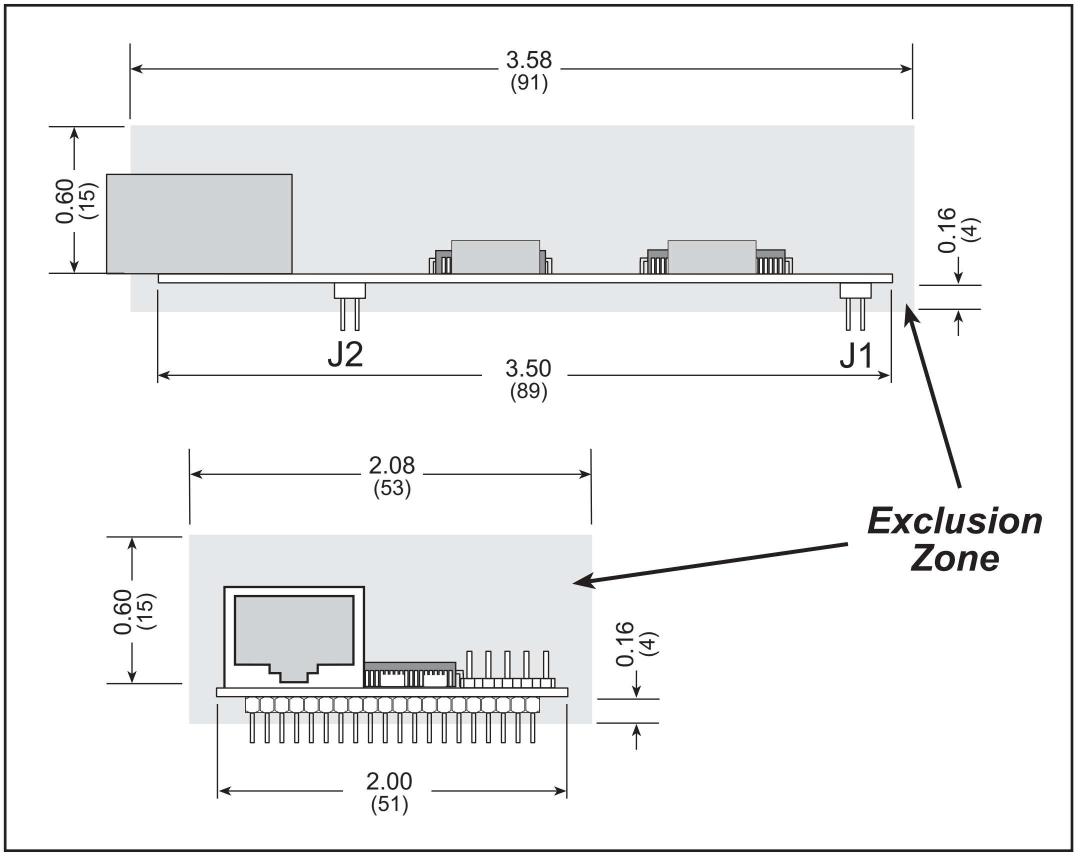

Figure A-2. RCM2100 “Exclusion Zone”

B.2 Mechanical Dimensions and Layout

Figure B-2 shows the mechanical dimensions and layout for the RCM2100 Prototyping Board.

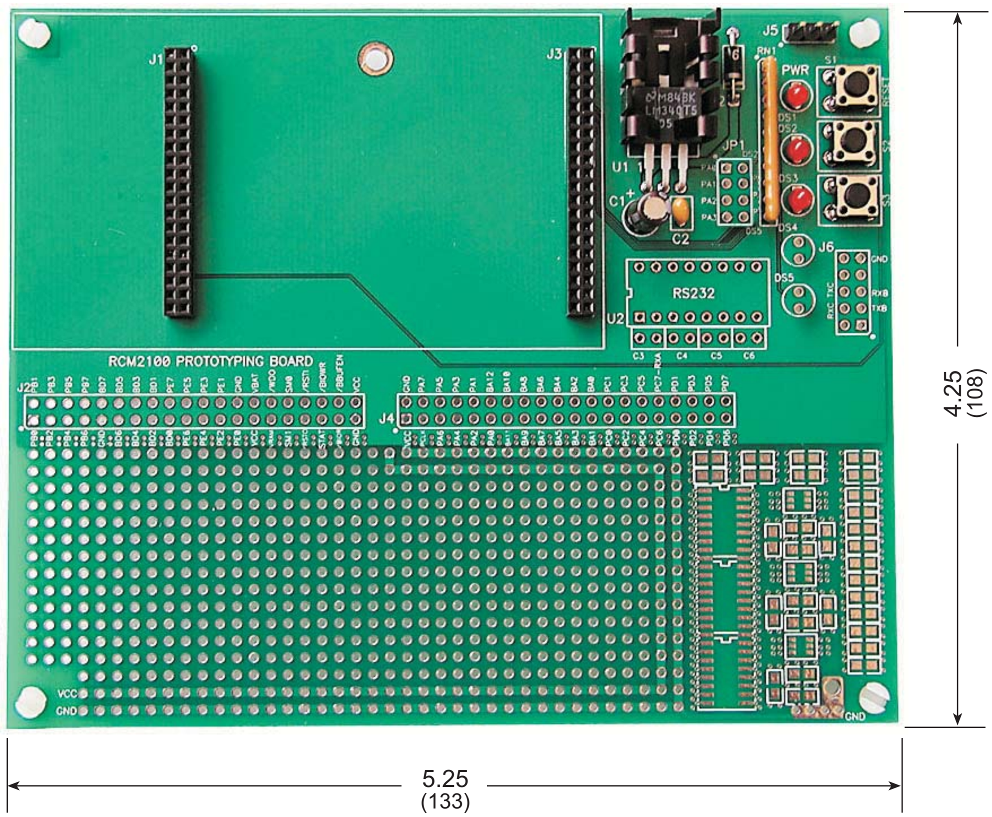

Figure B-2. RCM2100 Prototyping Board Dimensions

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

Table B-1. Prototyping Board Specifications

Parameter	Specification
Board Size	4.25" × 5.25" × 1.00" (108 mm × 133 mm × 25 mm)
Operating Temperature	–40°C to +70°C
Humidity	5% to 95%, noncondensing
Input Voltage	7.5 V to 25 V DC
Maximum Current Draw (including user-added circuits)	1 A at 12 V and 25°C, 0.7 A at 12 V and 70ºC
Prototyping Area	1.7" × 4" (43 mm × 102 mm) throughhole, 0.1" spacing
Standoffs/Spacers	4, accept 6-32 × 3/8 screws

B.3 Power Supply

The RCM2100 requires a regulated 5 V ± 0.25 V DC power source to operate. Depending on the amount of current required by the application, different regulators can be used to supply this voltage.

The Prototyping Board has an onboard LM340-T5 or equivalent. The LM340-T5 is an inexpensive linear regulator that is easy to use. Its major drawback is its inefficiency, which is directly proportional to the voltage drop across it. The voltage drop creates heat and wastes power.

A switching power supply may be used in applications where better efficiency is desirable. The LM2575 is an example of an easy-to-use switcher. This part greatly reduces the heat dissipation of the regulator. The drawback in using a switcher is the increased cost.

The Prototyping Board itself is protected against reverse polarity by a Shottky diode at D2 as shown in Figure B-3.

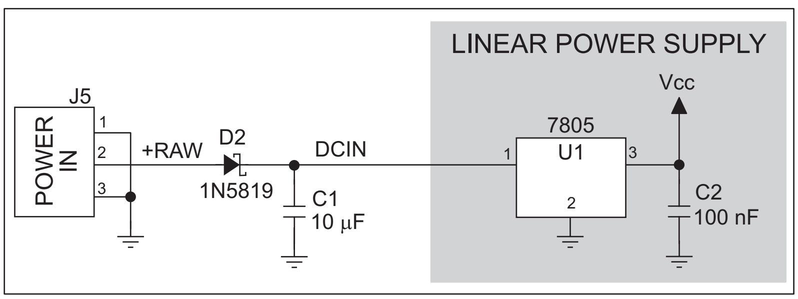

Figure B-3. Prototyping Board Power Supply

Capacitor C1 provides surge current protection for the voltage regulator, and allows the external power supply to be located some distance away.

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 RCM2100 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 RCM2100. 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 RCM2100.

Two more LEDs, driven by PA2 and PA3, may be added to the Prototyping Board for additional outputs.

To maximize the availability of RCM2100 resources, the demonstration hardware (LEDs and switches) on the Prototyping Board may be disconnected. This is done by cutting the traces below the silk-screen outline of header JP1 on the bottom side of the Prototyping Board. Figure B-4 shows the four places where cuts should be made. An exacto knife would work nicely to cut the traces. Alternatively, a small standard screwdriver may be carefully and forcefully used to wipe through the PCB traces.

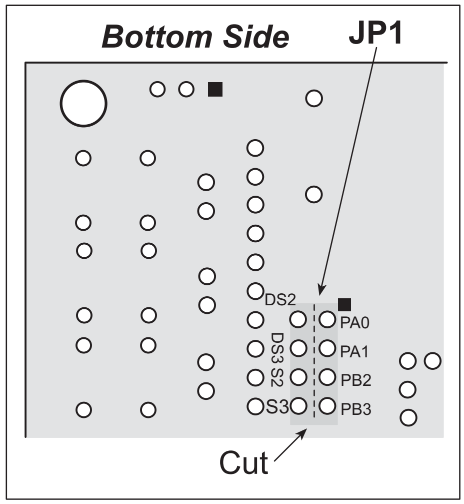

Figure B-4. Where to Cut Traces to Permanently Disable Demonstration Hardware on Prototyping Board

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

Table B-2. Prototyping Board Jumper Settings

Header JP2	
Pins	**Description**
1–2	PA0 to LED DS2
3–4	PA1 to LED DS3
5–6	PB2 to Switch S2
7–8	PB3 to Switch S3

Note that the pinout at location JP1 on the bottom side of the Prototyping Board (shown in Figure B-4) is a mirror image of the top side pinout.

The Prototyping Board provides the user with RCM2100 connection points brought out conveniently to labeled points at headers J2 and J4 on the Prototyping Board. Small to medium circuits can be prototyped using point-to-point wiring with 20 to 30 AWG wire between the prototyping area and the holes at locations J2 and J4. The holes are spaced at 0.1" (2.5 mm), and 40-pin headers or sockets may be installed at J2 and J4. The pinouts for locations J1 and J3, which correspond to J2 and J4, are shown in Figure B-5.

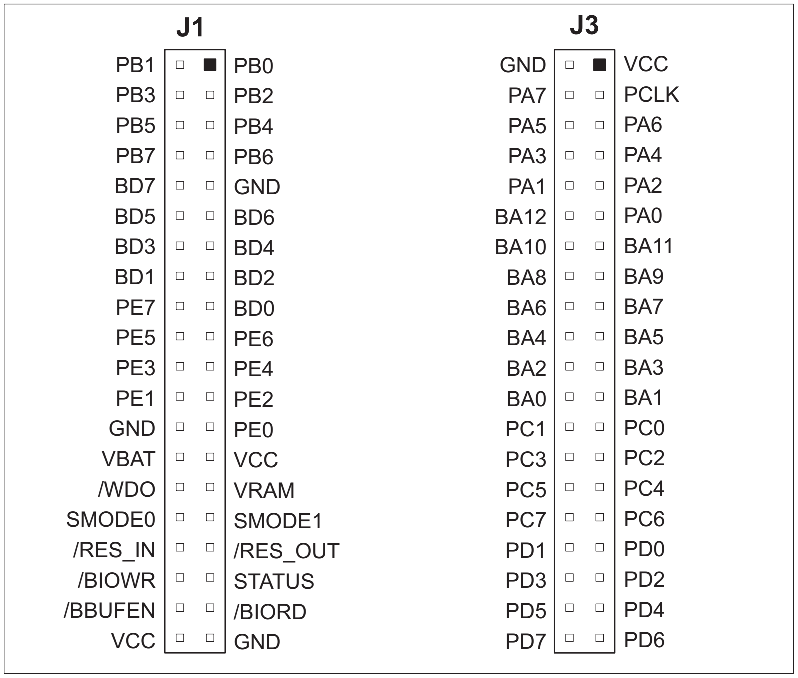

Figure B-5. RCM2100 Prototyping Board Pinout
(Top View)

A pair of small holes capable of holding 30 AWG wire appears below each hole pair at locations J2 and J4 for convenience in point-to-point wiring when headers are installed. The signals are those of the adjacent pairs of holes at J2 and J4. These small holes are also provided for the components that may be installed to the right of the prototyping area.

There is an additional 2" × 3" of through-hole prototyping space available on the Prototyping Board. VCC and GND traces run along the edge of the Prototyping Board for easy access. A GND pad is also provided at the lower right for alligator clips or probes.

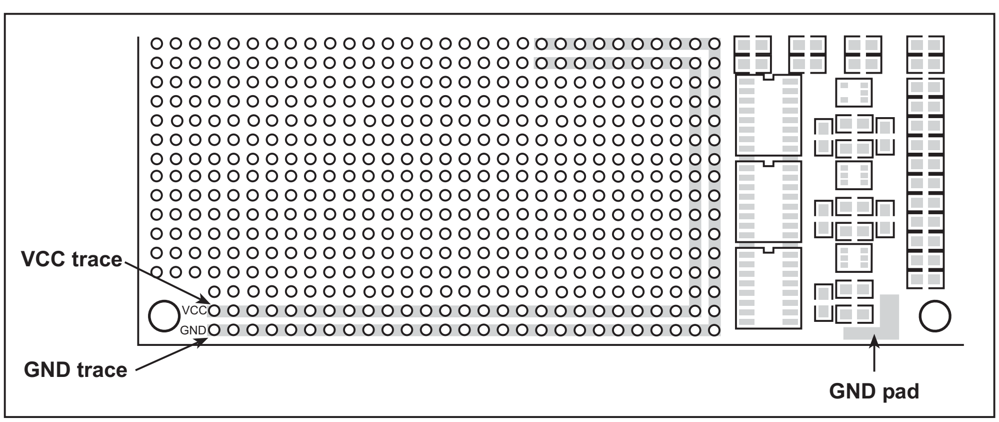

Figure B-6. VCC and GND Traces Along Edge of Prototyping Board

B.4.1 Adding Other Components

There is room on the Prototyping Board for a user-supplied RS-232 transceiver chip at location U2 and a 10-pin header for serial interfacing to external devices at location J6. A Maxim MAX232 transceiver is recommended. When adding the MAX232 transceiver at position U2, you must also add 100 nF charge storage capacitors at positions C3–C6 as shown in Figure B-7.

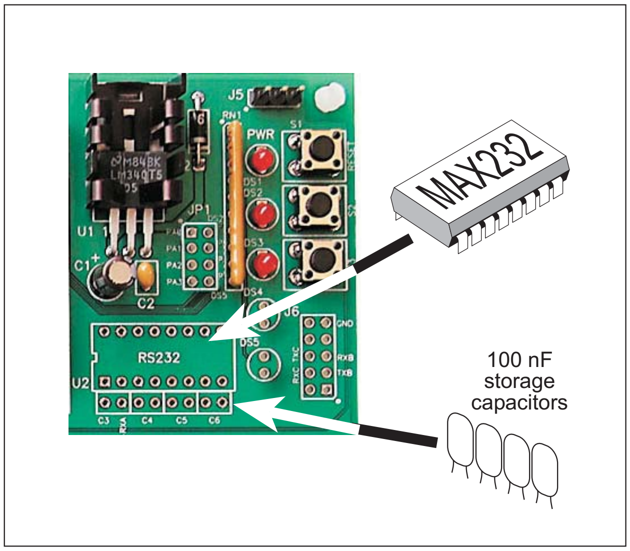

Figure B-7. Location for User-Supplied RS-232 Transceiver and Charge Storage Capacitors

There are two sets of pads that can be used for surface mount prototyping SOIC devices. The silk screen layout separates the rows into six 16-pin devices (three on each side). However, there are pads between the silk screen layouts giving the user two 52-pin (2×26) SOIC layouts with 50 mil pin spacing. There are six sets of pads that can be used for 3- to 6-pin SOT23 packages. There are also 60 sets of pads that can be used for SMT resistors and capacitors in an 0805 SMT package. Each component has every one of its pin pads connected to a hole in which a 30 AWG wire can be soldered (standard wire wrap wire can be soldered in for point-to-point wiring on the Prototyping Board). Because the traces are very thin, carefully determine which set of holes is connected to which surface-mount pad.

A lithium battery with a nominal voltage of 3 V and a minimum capacity of 165 mA·h is recommended. A lithium battery is strongly recommended because of its nearly constant nominal voltage over most of its life.

The drain on the battery by the RCM2100 is typically 16 µA when no other power is supplied. If a 950 mA·h battery is used, the battery can last more than 6 years:

$$\frac{950 \text{ mA·h}}{16 \text{ µA}} = 6.8 \text{ years.}$$

The actual life in your application will depend on the current drawn by components not on the RCM2100 and the storage capacity of the battery. Note that the shelf life of a lithium battery is ultimately 10 years.

The battery-backup circuit serves three purposes:

- It reduces the battery voltage to the SRAM and to the real-time clock, thereby limiting the current consumed by the real-time clock and lengthening the battery life.
- It ensures that current can flow only ***out*** of the battery to prevent charging the battery.
- A voltage, VOSC, is supplied to U7, which keeps the 32.768 kHz oscillator working when the voltage begins to drop.

VRAM and Vcc are nearly equal (<100 mV, typically 10 mV) when power is supplied to the RCM2100.

Figure C-2 shows the RCM2100 battery-backup circuit.

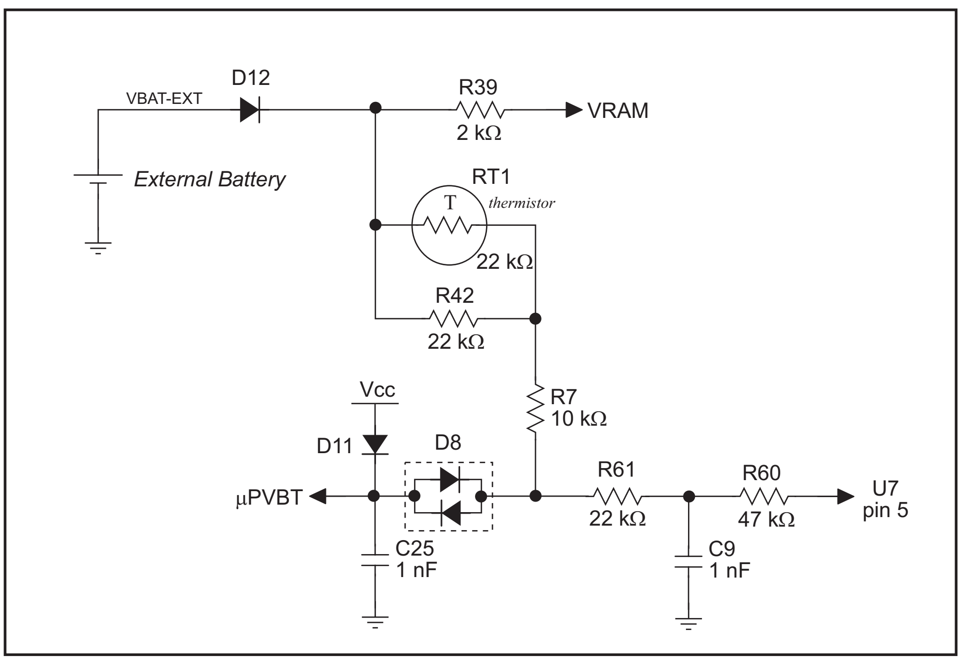

Figure C-2. RCM2100 Battery-Backup Circuit

D.4 D/A Converter

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

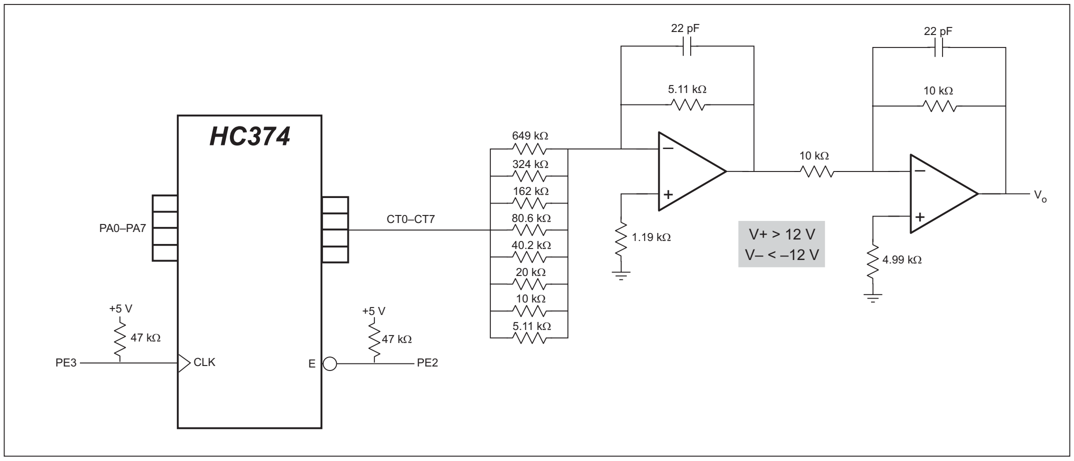

Figure D-5. Sample D/A Converter Connections

INDEX