**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 3000 |
| Co-Processor | - |
| Speed | 44.2MHz |
| Flash Size | 512KB (Internal), 4MB (External) |
| RAM Size | 1MB |
| Connector Type | 2 IDC Headers 2x17, 1 IDC Header 2x5 |
| Size / Dimension | 1.85" x 2.73" (47mm x 69mm) |
| Operating Temperature | -40°C ~ 70°C |
| Purchase URL | https://www.e-xfl.com/product-detail/digi-international/101-1068 |

# TABLE OF CONTENTS

### 2.2.3 Step 3 — Connect Power

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

If you have the universal power supply, prepare the AC adapter for the country where it will be used by selecting the plug. The RCM3305 Series 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 3(a), then press down on the spring-loaded clip below the plug assembly to allow the plug assembly to click into place.

Depending on the style of adapter, connect the AC adapter to 3-pin header J2 or jack J1 on the Prototyping Board as shown in Figure 3(a) or Figure 3(b).

Plug in the AC adapter. The red **CORE** LED on the Prototyping Board should light up. The RCM3305 series RabbitCore module 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 without disconnecting power.

#### 2.2.3.1 Alternate Power-Supply Connections

All Development Kits sold up to May, 2008, included a header connector that may be used to connect your power supply to 3-pin header J2 on the Prototyping Board. The connector may be attached either way as long as it is not offset to one side—the center pin of J2 is always connected to the positive terminal, and either edge pin is negative. The power supply should deliver 8 V to 30 V DC at 8 W.

# 3. RUNNING SAMPLE PROGRAMS

To develop and debug programs for the RCM3305/RCM3315 (and for all other Rabbit hardware), you must install and use Dynamic C.

## 3.1 Introduction

To help familiarize you with the RCM3305 and RCM3315 modules, Dynamic C includes several sample programs. Loading, executing and studying these programs will give you a solid hands-on overview of the RCM3305/RCM3315's capabilities, as well as a quick start using Dynamic C as an application development tool.

> **NOTE:** The sample programs assume that you have at least an elementary grasp of the C programming language. If you do not, see the introductory pages of the ***Dynamic C User's Manual*** for a suggested reading list.

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

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

1. Your RCM3305/RCM3315 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 RCM3305/RCM3315 to your PC.
4. Power must be applied to the RCM3305/RCM3315 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, then press function key **F9** to compile and run the program. The RCM3305/RCM3315 must be in Program Mode (see Figure 8) and must be connected to a PC using the programming cable.

# 4. HARDWARE REFERENCE

Chapter 4 describes the hardware components and principal hardware subsystems of the RCM3305/RCM3315 modules. Appendix A, "RCM3305/RCM3315 Specifications," provides complete physical and electrical specifications.

Figure 4 shows the Rabbit-based subsystems designed into the RCM3305/RCM3315.

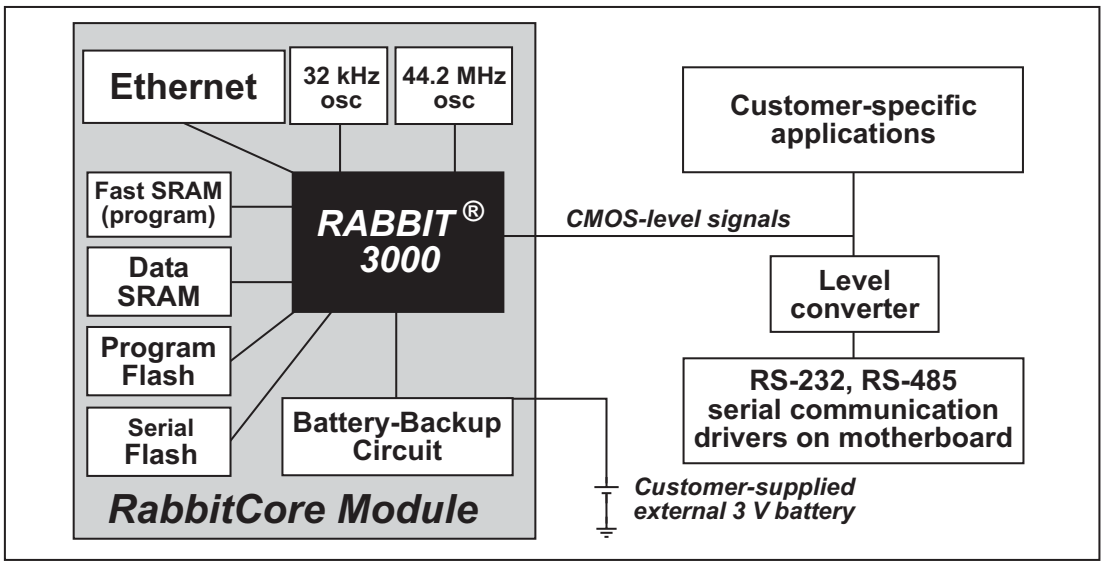


***Figure 4. RCM3305/RCM3315 Subsystems***

## 4.1 RCM3305/RCM3315 Digital Inputs and Outputs

Figure 5 shows the RCM3305/RCM3315 pinouts for headers J3 and J4.

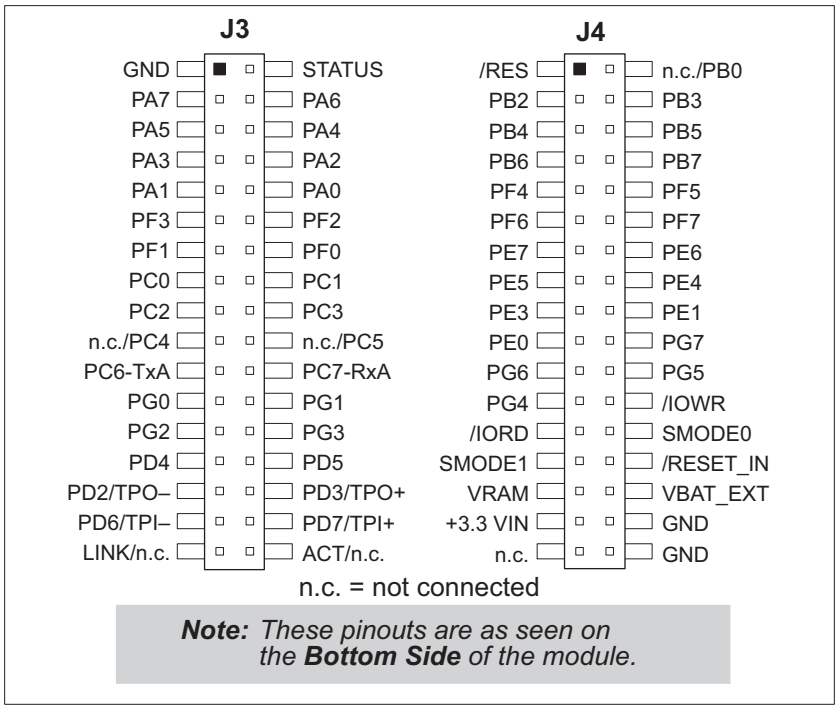


***Figure 5. RCM3305/RCM3315 Pinouts***

The pinouts for the RCM3000, RCM3100, RCM3200, RCM3305/RCM3315, RCM3360/RCM3370, and RCM3365/RCM3375 are almost compatible, except signals PB0, PC4, and PC5. PB0, PC4, and PC5 are used for the SPI interface to the serial flash on the RCM3305 and the RCM3315. Visit the Web site for further information.

Headers J3 and J4 are standard 2 × 34 headers with a nominal 2 mm pitch. An RJ-45 Ethernet port is also included with the RCM3305/RCM3315.

Pins 29–32 on header J3 are configured using 0 Ω resistors at locations JP4, JP5, JP6, and JP7 to be PD2, PD3, PD6, and PD7 respectively. They may also be reconfigured to carry the Ethernet signals TPI+, TPI–, TPO+, and TPO–.

Pins 33 and 34 on header J3 are wired to carry the **LINK** and **ACT** signals that illuminated the corresponding LEDs on the RCM3305/RCM3315 module. These signals may be "disconnected" by removing 0 Ω surface-mount resistors R41 and R42.

See Appendix A.5 for more information about the locations of these surface-mount resistors.

### 4.1.1 Memory I/O Interface

The Rabbit 3000 address lines (A0–A18) and all the data lines (D0–D7) are routed internally to the onboard flash memory and SRAM chips. I/0 write (/IOWR) and I/0 read (/IORD) are available for interfacing to external devices.

Parallel Port A can also be used as an external I/O data bus to isolate external I/O from the main data bus. Parallel Port B pins PB2–PB5 and PB7 can also be used as an external address bus.

When using the external I/O bus for a digital output or the LCD/keypad module on the Prototyping Board, or for any other reason, you must add the following line at the beginning of your program.

```
#define PORTA_AUX_IO    // required to enable external I/O bus
```

### 4.1.2 Other Inputs and Outputs

The status, /RESET_IN, SMODE0, and SMODE1 I/O are normally associated with the programming port. Since the status pin is not used by the system once a program has been downloaded and is running, the status pin can then be used as a general-purpose CMOS output. The programming port is described in more detail in Section 4.2.3.

/RES is an output from the reset circuitry that can be used to reset external peripheral devices.

### 4.1.3 LEDs

The RCM3305/RCM3315 has three Ethernet status LEDs located beside the RJ-45 Ethernet jack—these are discussed in Section 4.2.

Addiitionally, there are two other LEDs. The **SF** LED at DS3 blinks when data are being written to or read from the flash mass-storage device. The red **USR** LED at DS3 is a user-programmable LED, which is controlled by PD0 on the Rabbit 3000's Parallel Port D. The sample program **FLASHLED.C** provided in the Dynamic C **SAMPLES\RCM3300** folder shows how to set up and use this user-programmable LED.

### 4.2.3 Programming Port

The RCM3305/RCM3315 is programmed either through the serial programming port, which is accessed using header J1, or through the Ethernet jack. The RabbitLink may be used to provide a serial connection via the RabbitLink's Ethernet jack. The programming port uses the Rabbit 3000's Serial Port A for communication; Serial Port A is not used when programming is done over an Ethernet connection via the Dynamic C download manager or the remote application update. 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 3000 on the RCM3305/RCM3315 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.

In addition to Serial Port A, the Rabbit 3000 startup-mode (SMODE0, SMODE1), status, and reset pins are available on the programming port.

The two startup mode pins determine what happens after a reset—the Rabbit 3000 is either cold-booted or the program begins executing at address 0x0000.

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 CMOS output.

The /RESET_IN pin is an external input that is used to reset the Rabbit 3000 and the RCM3305/RCM3315 onboard peripheral circuits. The serial programming port can be used to force a hard reset on the RCM3305/RCM3315 by asserting the /RESET_IN signal.

#### Alternate Uses of the 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 I/O pin

The programming port may also be used as a serial port once the application is running. The SMODE pins may then be used as inputs and the status pin may be used as an output.

Refer to the ***Rabbit 3000 Microprocessor User's Manual*** for more information.

for additional information if you are using a Dynamic C release prior to v. 9.60 under Windows Vista. 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.
- 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.

# 6. USING THE TCP/IP FEATURES

## 6.1 TCP/IP Connections

Programming and development can be done with the RCM3305/RCM3315 modules without connecting the Ethernet port to a network. However, if you will be running the sample programs that use the Ethernet capability or will be doing Ethernet-enabled development, you should connect the RCM3305/RCM3315 module's Ethernet port at this time.

Before proceeding you will need to have the following items.

- If you don't have Ethernet access, you will need at least a 10Base-T Ethernet card (available from your favorite computer supplier) installed in a PC.
- Two RJ-45 straight-through Ethernet cables and a hub, or an RJ-45 crossover Ethernet cable.

A straight-through and a crossover Ethernet cable are included in both the RCM3305/RCM3315 Development Kit. Figure 9 shows how to identify the two cables based on the wires in the transparent RJ-45 connectors.

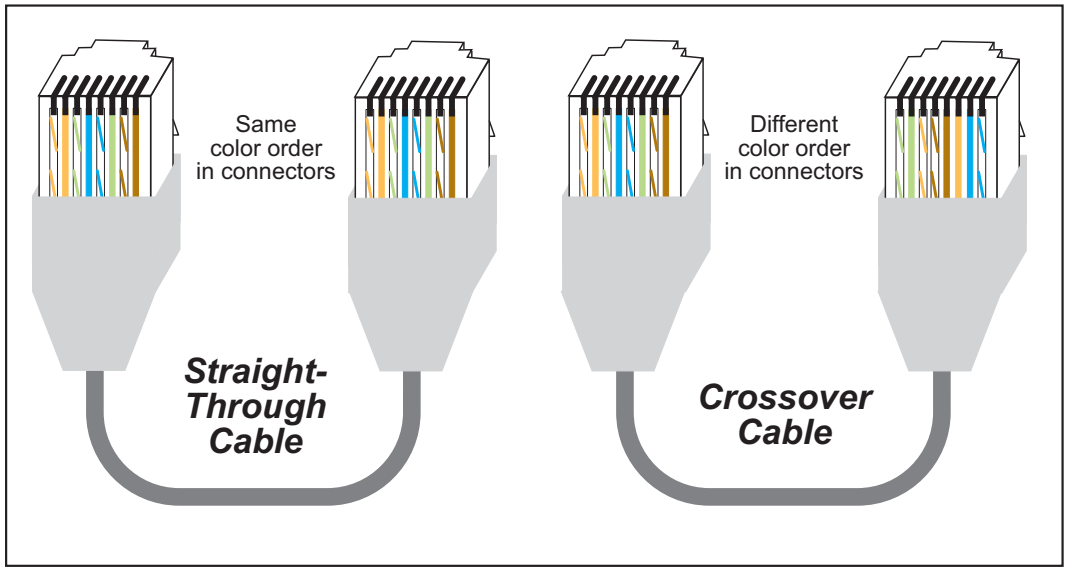


***Figure 9. How to Identify Straight-Through and Crossover Ethernet Cables***

Ethernet cables and a 10Base-T Ethernet hub are available from Rabbit in a TCP/IP tool kit. More information is available at www.rabbit.com.

# APPENDIX A. RCM3305/RCM3315 SPECIFICATIONS

Appendix A provides the specifications for the RCM3305/ RCM3315, and describes the conformal coating.

It is recommended that you allow for an “exclusion zone” of 0.04" (1 mm) around the RCM3305/RCM3315 in all directions when the RCM3305/RCM3315 is incorporated into an assembly that includes other printed circuit boards. An “exclusion zone” of 0.08" (2 mm) is recommended below the RCM3305/RCM3315 when the RCM3305/RCM3315 is plugged into another assembly. Figure A-2 shows this “exclusion zone.”

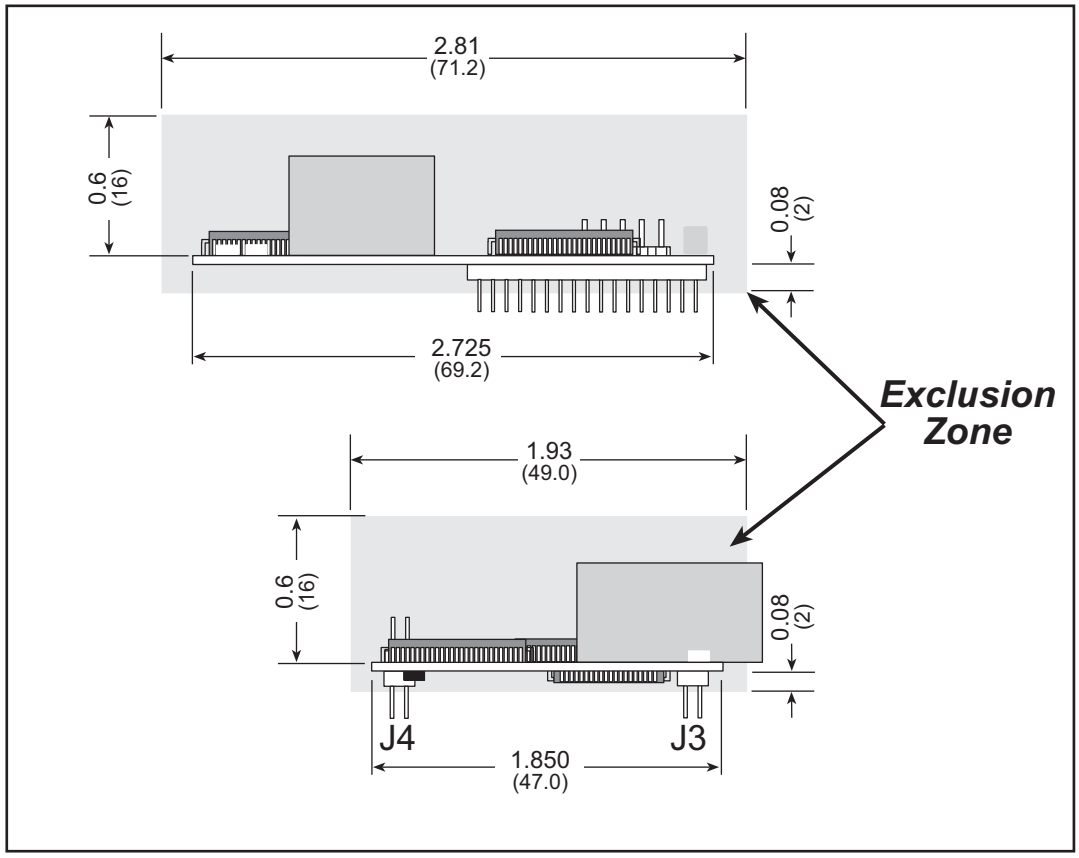


***Figure A-2. RCM3305/RCM3315 “Exclusion Zone”***

**NOTE:** All measurements are in inches followed by millimeters enclosed in parentheses.

Figure A-4 shows a typical timing diagram for the Rabbit 3000 microprocessor external I/O read and write cycles.

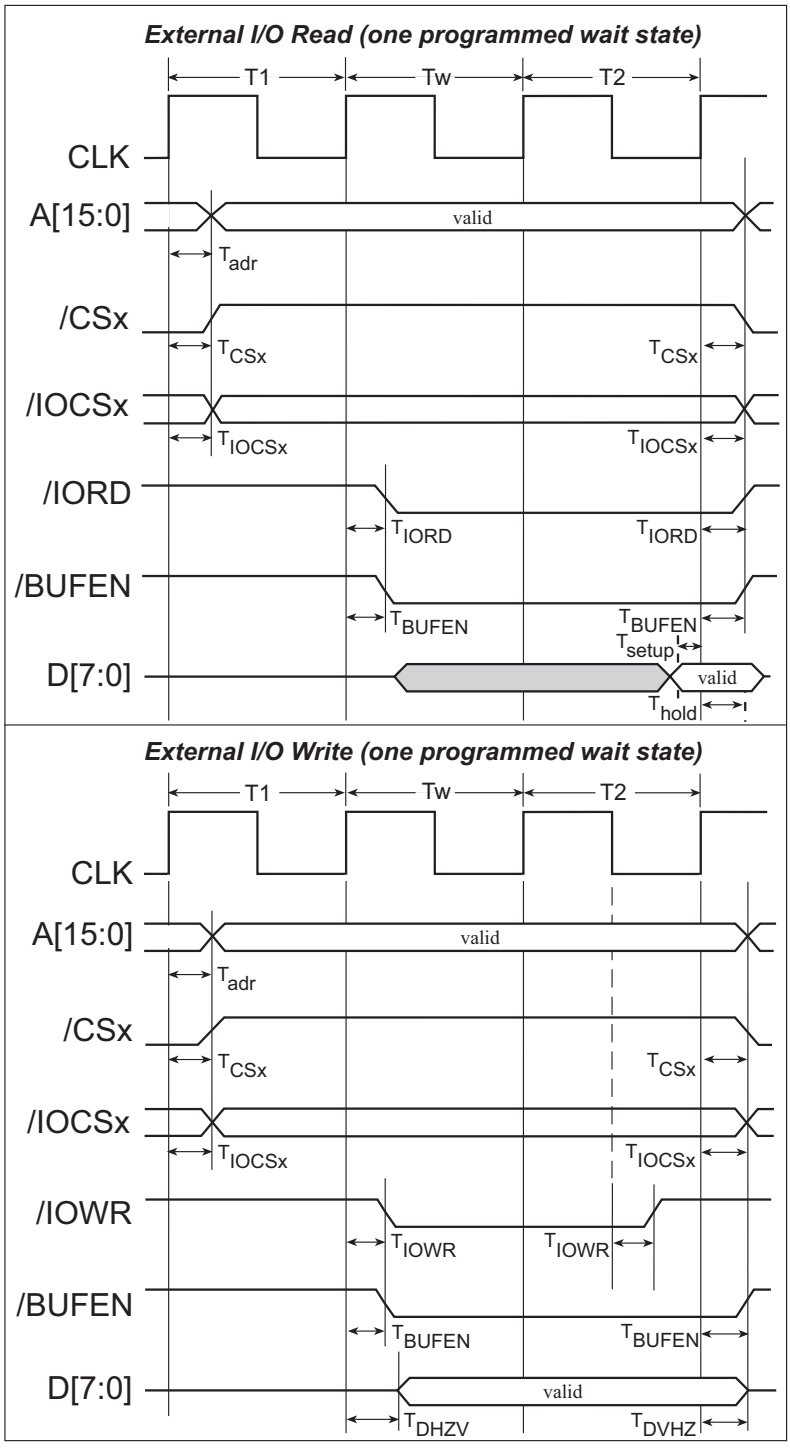


***Figure A-4. I/O Read and Write Cycles—No Extra Wait States***

**NOTE:** ***/IOCSx*** can be programmed to be active low (default) or active high.

### B.4.3 CMOS Digital Outputs

If the stepper-motor option is not used, eight CMOS-level digital outputs are available at J10, and can each handle up to 25 mA.

### B.4.4 Sinking Digital Outputs

Four sinking digital outputs shared with LEDs DS3–DS6 are available at J12, and can each handle up to 500 mA. Figure B-6 shows a wiring diagram for a typical sinking output.

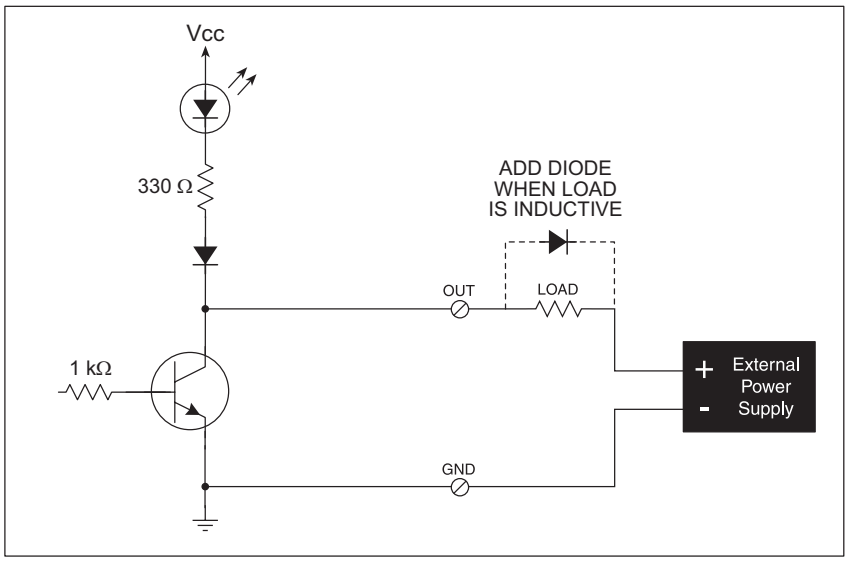


***Figure B-6. Prototyping Board Sinking Digital Outputs***

### B.4.5 Relay Outputs

Figure B-7 shows the contact connections for the relay on the Prototyping Board. A diode across the coil provides a return path for inductive spikes, and snubbers across the relay contacts protect the relay contacts from inductive spikes.

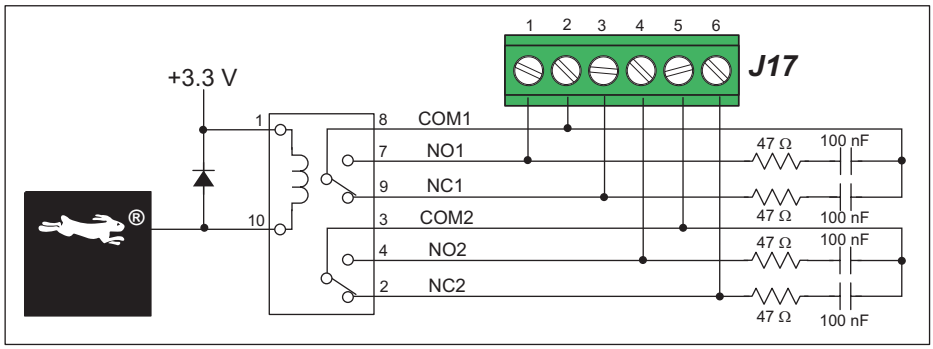


***Figure B-7. Prototyping Board Relay Output Contact Connections***

The relay is driven by pin PA4 of the RCM3305/RCM3315 module via U8, and is controlled by PE7 and PG5 as shown in the sample applications.

The Prototyping Board comes with a 220 Ω termination resistor and two 681 Ω bias resistors installed and enabled with jumpers across pins 1–2 and 5–6 on header JP5, as shown in Figure B-9.

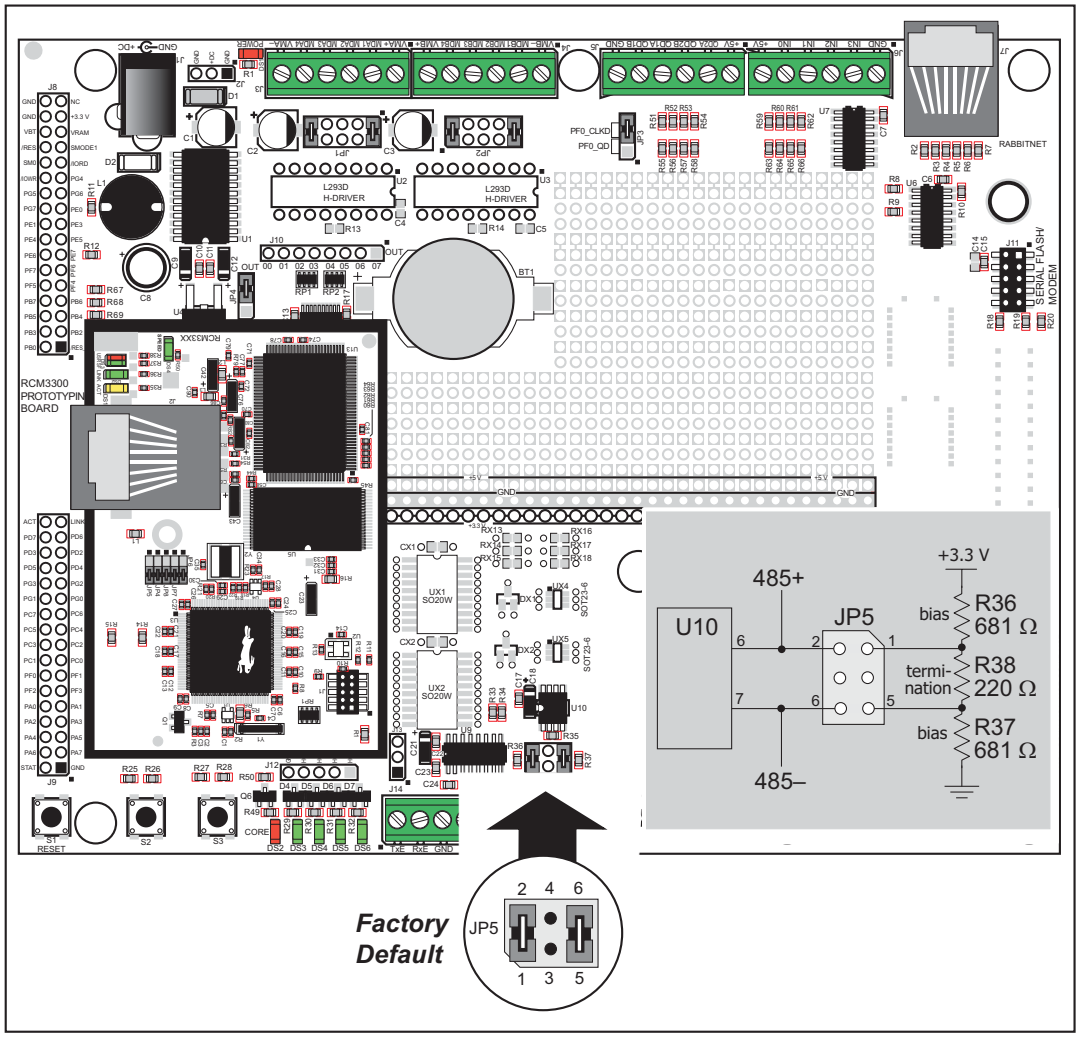


***Figure B-9. RS-485 Termination and Bias Resistors***

For best performance, the termination resistors in a multidrop network should be enabled only on the end nodes of the network, but ***not*** on the intervening nodes. Jumpers on boards whose termination resistors are not enabled may be stored across pins 1–3 and 4–6 of header JP5.

### B.4.7 RabbitNet Ports

The RJ-45 jack labeled ***RabbitNet*** is a clocked SPI RS-422 serial I/O expansion port for use with RabbitNet peripheral boards. The ***RabbitNet*** jack does ***not*** support Ethernet connections. Header JP3 must have pins 2–3 jumpered when using the RabbitNet port.

The RabbitNet port is enabled in software by setting PD2 = 1. Note that the RabbitNet port and the J11 interface cannot be used simultaneously.

## C.2 Contrast Adjustments for All LCD/Keypad Modules

Starting in 2005, LCD/keypad modules were factory-configured to optimize their contrast based on the voltage of the system they would be used in. Be sure to select a KDU3V LCD/keypad module for use with the Prototyping Board for the RCM3305/RCM3315 — these modules operate at 3.3 V. You may adjust the contrast using the potentiometer at R2 as shown in Figure C-3. LCD/keypad modules configured for 5 V may be used with the 3.3 V RCM3300 Prototyping Board, but the backlight will be dim.

***LCD/Keypad Module Jumper Configurations***

| Header | Description | Pins Connected | Factory Default |
|---|---|---|---|
| J5 | 2.8 V | 1–2 | × |
| | 3.3 V | 3–4 | |
| | 5 V | n.c. | |

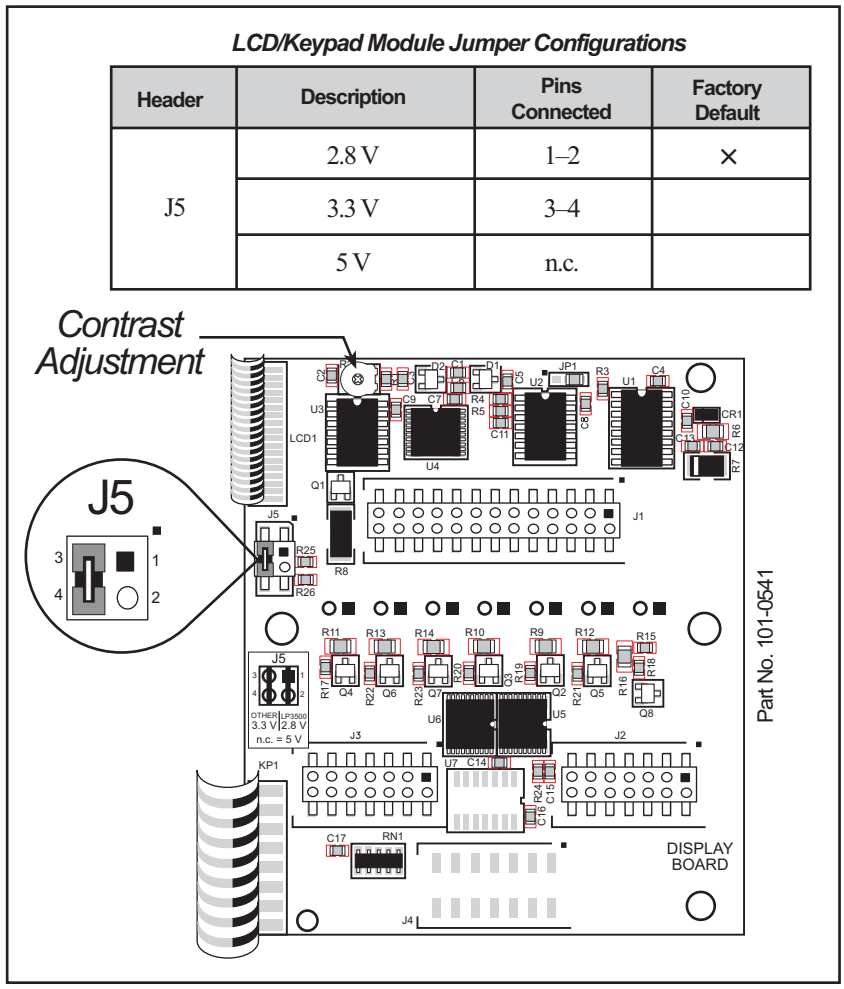


***Figure C-3. LCD/Keypad Module Contrast Adjustments***

You can set the contrast on the LCD display of pre-2005 LCD/keypad modules by adjusting the potentiometer at R2 or by setting the voltage for 3.3 V by connecting the jumper across pins 3–4 on header J5 as shown in Figure C-3. Only one of these two options is available on these LCD/keypad modules.

**NOTE:** Older LCD/keypad modules that do not have a header at J5 or a contrast adjustment potentiometer at R2 are limited to operate only at 5 V, and will not work with the Prototyping Board for the RCM3305/RCM3315. The older LCD/keypad modules are no longer being sold.

## C.7 Sample Programs

Sample programs illustrating the use of the LCD/keypad module with the Prototyping Board are provided in the **SAMPLES\RCM3300\LCD_KEYPAD** folder.

These sample programs use the external I/O bus on the Rabbit 3000 chip, and so the **#define PORTA_AUX_IO** line is already included in the sample programs.

Each sample program has comments that describe the purpose and function of the program. Follow the instructions at the beginning of the sample program. 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**. The RCM3305/RCM3315 must be connected to a PC using the programming cable as described in Chapter 2, "Getting Started."

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

- **KEYPADTOLED.C**—This program demonstrates the use of the external I/O bus. The program will light up an LED on the LCD/keypad module and will display a message on the LCD when a key press is detected. The DS3, DS4, DS5, and DS6 LEDs on the Prototyping Board will also light up.
- **LCDKEYFUN.C**—This program demonstrates how to draw primitive features from the graphic library (lines, circles, polygons), and also demonstrates the keypad with the key release option.
- **SWITCHTOLCD.C**—This program demonstrates the use of the external I/O bus. The program will light up an LED on the LCD/keypad module and will display a message on the LCD when a switch press is detected. The DS1 and DS2 LEDs on the Prototyping Board will also light up.

Additional sample programs are available in the **SAMPLES\LCD_KEYPAD\122x32_1x7** folder.

```
void glHScroll(int left, int top, int cols,
   int rows, int nPix);
```

Scrolls right or left, within the defined window by *x* number of pixels. The opposite edge of the scrolled window will be filled in with white pixels. The window must be byte-aligned.

Parameters will be verified for the following:

1. The **`left`** and **`cols`** parameters will be verified that they are evenly divisible by 8. If not, they will be truncated to a value that is a multiple of 8.
2. Parameters will be checked to verify that the scrolling area is valid. The minimum scrolling area is a width of 8 pixels and a height of one row.

#### PARAMETERS

**`left`** is the top left corner of bitmap, must be evenly divisible by 8.

**`top`** is the top left corner of the bitmap.

**`cols`** is the number of columns in the window, must be evenly divisible by 8.

**`rows`** is the number of rows in the window.

**`nPix`** is the number of pixels to scroll within the defined window (a negative value will produce a scroll to the left).

#### RETURN VALUE

None.

#### SEE ALSO

**`glVScroll`**

**cSpdHi** is a high-speed repeat tick, which is approximately one debounce period or 5 µs.

How many times to repeat after low speed repeat.
0 = None.

**RETURN VALUE**

None.

**SEE ALSO**

**keyProcess, keyGet, keypadDef**

## void keyProcess(void);

Scans and processes keypad data for key assignment, debouncing, press and release, and repeat.

**NOTE:** This function is also able to process an 8 × 8 matrix keypad.

**RETURN VALUE**

None

**SEE ALSO**

**keyConfig, keyGet, keypadDef**

## char keyGet(void);

Get next keypress.

**RETURN VALUE**

The next keypress, or 0 if none

**SEE ALSO**

**keyConfig, keyProcess, keypadDef**

## int keyUnget(char cKey);

Pushes the value of **cKey** to the top of the input queue, which is 16 bytes deep.

**PARAMETER**

**cKey**

**RETURN VALUE**

None.

**SEE ALSO**

**keyGet**

Use a straight-through Ethernet cable to connect the master to slave peripheral cards, unless you are using a device such as the OP7200 that could be used either as a master or a slave. In this case you would use a crossover cable to connect an OP7200 that is being used as a slave.

Distances between a master unit and peripheral cards can be up to 10 m or 33 ft.

### E.1.2 RabbitNet Peripheral Cards

- Digital I/O

  24 inputs, 16 push/pull outputs, 4 channels of 10-bit A/D conversion with ranges of 0 to 10 V, 0 to 1 V, and -0.25 to +0.25 V. The following connectors are used:

  Signal = 0.1" friction-lock connectors
  Power = 0.156" friction-lock connectors
  RabbitNet = RJ-45 connector

- A/D converter

  8 channels of programmable-gain 12-bit A/D conversion, configurable as current measurement and differential-input pairs. 2.5 V reference voltage is available on the connector. The following connectors are used:

  Signal = 0.1" friction-lock connectors
  Power = 0.156" friction-lock connectors
  RabbitNet = RJ-45 connector

- D/A converter

  8 channels of 0–10 V 12-bit D/A conversion. The following connectors are used:

  Signal = 0.1" friction-lock connectors
  Power = 0.156" friction-lock connectors
  RabbitNet = RJ-45 connector

- Display/Keypad interface

  allows you to connect your own keypad with up to 64 keys and one character liquid crystal display from 1 × 8 to 4 × 40 characters with or without backlight using standard 1 × 16 or 2 × 8 connectors. The following connectors are used:

  Signal = 0.1" headers or sockets
  Power = 0.156" friction-lock connectors
  RabbitNet = RJ-45 connector

- Relay card

  6 relays rated at 250 V AC, 1200 V·A or 100 V DC up to 240 W. The following connectors are used:

  Relay contacts = screw-terminal connectors
  Power = 0.156" friction-lock connectors
  RabbitNet = RJ-45 connector

Visit our Web site for up-to-date information about additional cards and features as they become available. The Web site also has the latest revision of this user's manual.