Digi - 20-101-1051 Datasheet

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	Not For New Designs
Module/Board Type	MPU Core
Core Processor	Rabbit 3000
Co-Processor	-
Speed	44.2MHz
Flash Size	512KB (Internal), 32MB (External)
RAM Size	1MB
Connector Type	2 IDC Headers 2x17, 1 IDC Header 2x5. 1 xD-Picture Card
Size / Dimension	1.85" x 2.73" (47mm x 69mm)
Operating Temperature	0°C ~ 70°C
Purchase URL	https://www.e-xfl.com/product-detail/digi-international/20-101-1051

B.3 Power Supply 87
B.4 Using the Prototyping Board 88
B.4.1 Adding Other Components 89
B.4.2 Digital I/O 90
B.4.2.1 Digital Inputs 90
B.4.3 CMOS Digital Outputs 91
B.4.4 Sinking Digital Outputs 91
B.4.5 Relay Outputs 91
B.4.6 Serial Communication 92
B.4.6.1 RS-232 93
B.4.6.2 RS-485 94
B.4.7 RabbitNet Ports 95
B.4.8 Other Prototyping Board Modules 96
B.4.9 Quadrature Decoder 96
B.4.10 Stepper-Motor Control 96
B.5 Prototyping Board Jumper Configurations 98
B.6 Use of Rabbit 3000 Parallel Ports 100

Appendix C. LCD/Keypad Module 103

C.1 Specifications 103
C.2 Contrast Adjustments for All Boards 105
C.3 Keypad Labeling 106
C.4 Header Pinouts 107
C.4.1 I/O Address Assignments 107
C.5 Mounting LCD/Keypad Module on the Prototyping Board 108
C.6 Bezel-Mount Installation 109
C.6.1 Connect the LCD/Keypad Module to Your Prototyping Board 111
C.7 Sample Programs 112
C.8 LCD/Keypad Module Function Calls 113
C.8.1 LCD/Keypad Module Initialization 113
C.8.2 LEDs 113
C.8.3 LCD Display 114
C.8.4 Keypad 130

Appendix D. Power Supply 133

D.1 Power Supplies 133
D.1.1 Battery Backup 133
D.1.2 Battery-Backup Circuit 134
D.1.3 Reset Generator 135

Appendix E. Programming via Ethernet Crossover Cable 137

E.1 Load TCP/IP Parameters to the RCM3365 Module 138
E.2 Load TCP/IP Parameters to the PC, Notebook, or Workstation 139
E.3 Run a Program 141
E.3.1 Troubleshooting 141

Appendix F. RabbitNet 143

F.1 General RabbitNet Description 143
F.1.1 RabbitNet Connections 143
F.1.2 RabbitNet Peripheral Cards 144
F.2 Physical Implementation 145
F.2.1 Control and Routing 145
F.3 Function Calls 146
F.3.1 Status Byte 152

Index 153

1. Introduction

The RCM3365 and RCM3375 RabbitCore modules feature a compact module that incorporates the latest revision of the powerful Rabbit® 3000 microprocessor, flash memory, mass storage (NAND flash), static RAM, and digital I/O ports. The RCM3365 and RCM3375 present a new form of embedded flexibility with removable ("hot-swappable") memory cards. The RCM3365 and RCM3375 both have an integrated 10/100Base-T Ethernet port, and provide for LAN and Internet-enabled systems to be built as easily as serial-communication systems.

In addition to the features already mentioned above, the RCM3365 and RCM3375 have two clocks (main oscillator and real-time clock), reset circuitry, and the circuitry necessary for management of battery backup of the Rabbit 3000's internal real-time clock and the static RAM. Two 34-pin headers bring out the Rabbit 3000 I/O bus lines, parallel ports, and serial ports.

The RCM3365/RCM3375's mass-storage capabilities make them suited to running the optional Dynamic C FAT file system module where data are stored and handled using the same directory file structure commonly used on PCs. A removable *xD-Picture Card* can be hot-swapped to transfer data quickly and easily using a standardized file system that can be read away from the RCM3365/RCM3375 installation.

The RCM3365 or RCM3375 receives +3.3 V power from the customer-supplied motherboard on which it is mounted. The RCM3365 and RCM3375 can interface with all kinds of CMOS-compatible digital devices through the motherboard.

The Development Kit has what you need to design your own microprocessor-based system: a complete Dynamic C software development system including the Dynamic C FAT File System module, and a Prototyping Board that allows you to evaluate the RCM3365 or RCM3375, and to prototype circuits that interface to the RCM3365 or RCM3375 module.

1.1 RCM3365 and RCM3375 Features

- Small size: 1.85" x 2.73" x 0.86"
  (47 mm x 69 mm x 22 mm)
- Microprocessor: Rabbit 3000 running at 44.2 MHz
- 52 parallel 5 V tolerant I/O lines: 44 configurable for I/O, 4 fixed inputs, 4 fixed outputs
- Three additional digital inputs, two additional digital outputs
- External reset
- Alternate I/O bus can be configured for 8 data lines and 6 address lines (shared with parallel I/O lines), plus I/O read/write
- Ten 8-bit timers (six cascadable) and one 10-bit timer with two match registers
- 512K flash memory, 512K program execution SRAM, 512K data SRAM
- Fixed and hot-swappable mass-storage flash-memory options, which may be used with the standardized directory structure supported by the Dynamic C FAT File System module.
- Real-time clock
- Watchdog supervisor
- Provision for customer-supplied backup battery via connections on header J4
- 10-bit free-running PWM counter and four pulse-width registers
- Two-channel Input Capture (shared with parallel I/O ports) can be used to time input signals from various port pins
- Two-channel Quadrature Decoder accepts inputs from external incremental encoder modules
- Five or six 3.3 V CMOS-compatible serial ports with a maximum asynchronous baud rate of 5.525 Mbps. Three ports are configurable as a clocked serial port (SPI), and two ports are configurable as SDLC/HDLC serial ports (shared with parallel I/O ports).
- Supports 1.15 Mbps IrDA transceiver
- Supports Dynamic C RabbitSys, which supports Ethernet access for remote application updates, and remote monitoring and control of a RabbitSys-enabled RCM3365

The RCM3900/RCM3910 and RCM3365/RCM3375 RabbitCore modules are similar to the RCM3305/RCM3315 and RCM3309/RCM3319, but they use fixed NAND or removable media for their mass-storage memories instead of the fixed serial flash options of the RCM3305/RCM3315 and the RCM3309/RCM3319.

- **Ethernet chip** — A different Ethernet controller chip is used on the RCM3900/RCM3910. The Ethernet chip is able to detect automatically whether a crossover cable or a straight-through cable is being used in a particular setup, and will configure the signals on the Ethernet jack interface.
- **Dynamic C** — As long as no low-level FAT file system calls or direct *xD-Picture Card* access calls to the **`NFLASH.LIB`** library were used in your application developed for the RCM3365/RCM3375, you may run that application on the RCM3900/RCM3910 after you recompile it using Dynamic C v. 9.60.

  NOTE: The Dynamic C RabbitSys option for programming an RCM3365 over an Ethernet link is not supported for the RCM3900.

1.3 Advantages of the RCM3365 and RCM3375

- Fast time to market using a fully engineered, "ready-to-run/ready-to-program" microprocessor core.
- Competitive pricing when compared with the alternative of purchasing and assembling individual components.
- Easy C-language program development and debugging
- Program download utility (Rabbit Field Utility) and cloning board options for rapid production loading of programs.
- Generous memory size allows large programs with tens of thousands of lines of code, and substantial data storage.
- Integrated Ethernet port for network connectivity, with royalty-free TCP/IP software.
- Ideal for network-enabling security and access systems, home automation, HVAC systems, and industrial controls

4.2.3 Serial Programming Port

The RCM3365/RCM3375 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 on a board running Dynamic C RabbitSys. 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 RCM3365/RCM3375 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 RCM3365/RCM3375 onboard peripheral circuits. The serial programming port can be used to force a hard reset on the RCM3365/RCM3375 by asserting the /RESET_IN signal. No equivalent functionality exists for programming over an Ethernet connection on a board running Dynamic C RabbitSys.

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 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.

4.5 Other Hardware

4.5.1 Clock Doubler

The RCM3365/RCM33610 takes advantage of the Rabbit 3000 microprocessor's internal clock doubler. A built-in clock doubler allows half-frequency crystals to be used to reduce radiated emissions. The 44.2 MHz frequency specified for the RCM3365/RCM3375 is generated using a 22.12 MHz resonator.

The clock doubler may be disabled if 44.2 MHz clock speeds are not required. This 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.
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.

4.5.2 Spectrum Spreader

The Rabbit 3000 features a spectrum spreader, which helps to mitigate EMI problems. The spectrum spreader is on by default, but it may also be turned off or set to a stronger setting. The means for doing so is through a simple configuration macro as shown below.

1. Select the "Defines" tab from the Dynamic C **Options > Project Options** menu.
2. Normal spreading is the default, and usually no entry is needed. If you need to specify normal spreading, add the line

   ```
   ENABLE_SPREADER=1
   ```

   For strong spreading, add the line

   ```
   ENABLE_SPREADER=2
   ```

   To disable the spectrum spreader, add the line

   ```
   ENABLE_SPREADER=0
   ```

   NOTE: The strong spectrum-spreading setting is not recommended since it may limit the maximum clock speed or the maximum baud rate. It is unlikely that the strong setting will be used in a real application.
3. Click **OK** to save the macro. The spectrum spreader will now be set to the state specified by the macro value whenever you are in the project file where you defined the macro.

NOTE: Refer to the ***Rabbit 3000 Microprocessor User's Manual*** for more information on the spectrum-spreading setting and the maximum clock speed.

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 controllers and other controllers based on the Rabbit microprocessor. Chapter 5 describes the libraries and function calls related to the RCM3365/RCM3375.

5.1 More About Dynamic C

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

You have a choice of doing your software development in the flash memory or in the static SRAM included on the RCM3365/RCM3375. The flash memory and SRAM options are selected with the **Options > Program 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 should be run from the program execution SRAM after the serial programming cable is disconnected. Your final code must always be stored in flash memory for reliable operation. RCM3365/RCM3375 modules running at 44.2 MHz have a fast program execution SRAM that is not battery-backed. Select **Code and BIOS in Flash, Run in RAM** from the Dynamic C **Options > Project Options > Compiler** menu to store the code in flash and copy it to the fast program execution SRAM at run-time to take advantage of the faster clock speed. This option optimizes the performance of RCM3365/RCM3375 modules running at 44.2 MHz.

NOTE: Do not depend on the flash memory sector size or type in your program logic. The RCM3365/RCM3375 and Dynamic C were designed to accommodate flash devices with various sector sizes in response to the volatility of the flash-memory market.

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 2000/NT and later—see Rabbit's Technical Note TN257, *Running Dynamic C® With Windows Vista®*,

5.1.1 Developing Programs Remotely with Dynamic C

Dynamic C is an integrated development environment that allows you to edit, compile, and debug your programs. Dynamic C has the ability to allow programming over the Internet or local Ethernet. This is accomplished in one of three ways.

1. RCM3365 RabbitCore modules that are preloaded with Dynamic C RabbitSys firmware can be used with Dynamic C RabbitSys to be accessed via an Ethernet connection for remote application updates, and for remote monitoring and control. Dynamic C RabbitSys requires Dynamic C version 9.30 or later, and allows the RCM3365. The ***Dynamic C RabbitSys User's Manual*** provides complete information on RabbitSys.
2. Via the Rabbit RabbitLink, which allows a Rabbit-based target to have programs downloaded to it and debugged with the same ease as exists when the target is connected directly to a PC.
3. Dynamic C provides sample programs to illustrate the use of a download manager, but these sample programs are not intended for use with the NAND flash on the RCM3365 and RCM3375 RabbitCore modules. The **`DLM_TCP.C`** and **`DLP_TCP.C`** sample programs found in the Dynamic C **`SAMPLES\DOWN_LOAD`** folder, are intended to be compiled to the program flash memory (which is a parallel flash memory). Custom applications based on these sample programs may use the NAND flash for data storage.

5.2.6.3 Switches, LEDs, and Relay

```
int switchIn(int swin);
```

Reads the state of a switch input.

PARAMETERS

`swin` is the switch input to read:

2—S2
3—S3

RETURN VALUE

State of the switch input:

1 = open
0 = closed

SEE ALSO

`brdInit`

```
void ledOut(int led, int value);
```

Controls LEDs on the Prototyping Board and on the RCM3365/RCM3375.

PARAMETERS

`led` is the LED to control:

0 = red User LED on RCM3365/RCM3375
3 = DS3 on Prototyping Board
4 = DS4 on Prototyping Board
5 = DS5 on Prototyping Board
6 = DS6 on Prototyping Board

`value` is the value used to control the LED:

0 = off
1 = on

RETURN VALUE

None.

SEE ALSO

`brdInit`

Now you should be able to make your connections.

1. Connect the AC adapter and the serial programming cable as shown in Chapter 2, “Getting Started.”
2. Ethernet Connections

   There are four options for connecting the RCM3365/RCM3375 module to a network for development and runtime purposes. The first two options permit total freedom of action in selecting network addresses and use of the “network,” as no action can interfere with other users. We recommend one of these options for initial development.

- **No LAN —** The simplest alternative for desktop development. Connect the RCM3365/RCM3375 module’s Ethernet port directly to the PC’s network interface card using an RJ-45 *crossover cable*. A crossover cable is a special cable that flips some connections between the two connectors and permits direct connection of two client systems. A standard RJ-45 network cable will not work for this purpose.
- **Micro-LAN —** Another simple alternative for desktop development. Use a small Ethernet 10Base-T hub and connect both the PC’s network interface card and the RCM3365/RCM3375 module’s Ethernet port to it using standard network cables.

  The following options require more care in address selection and testing actions, as conflicts with other users, servers and systems can occur:

- **LAN —** Connect the RCM3365/RCM3375 module’s Ethernet port to an existing LAN, preferably one to which the development PC is already connected. You will need to obtain IP addressing information from your network administrator.
- **WAN —** The RCM3365/RCM3375 is capable of direct connection to the Internet and other Wide Area Networks, but exceptional care should be used with IP address settings and all network-related programming and development. We recommend that development and debugging be done on a local network before connecting a RabbitCore system to the Internet.

  TIP: Checking and debugging the initial setup on a micro-LAN is recommended before connecting the system to a LAN or WAN.

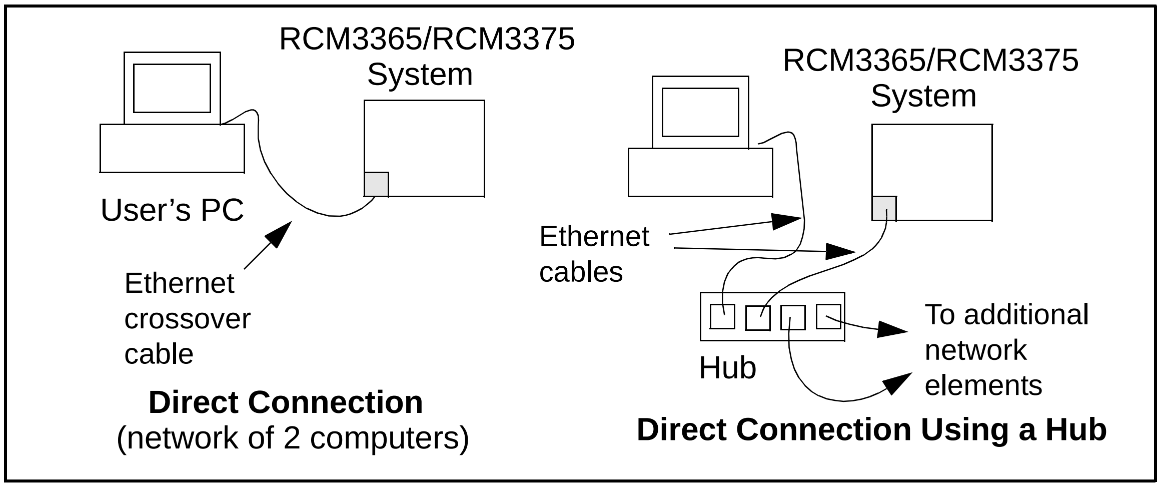
  The PC running Dynamic C does not need to be the PC with the Ethernet card.

3. Apply Power

   Plug in the AC adapter. The RCM3365/RCM3375 module and Prototyping Board are now ready to be used.

6.4 Running TCP/IP Sample Programs

We have provided a number of sample programs demonstrating various uses of TCP/IP for networking embedded systems. These programs require you to connect your PC and the RCM3365/RCM3375 board together on the same network. This network can be a local private network (preferred for initial experimentation and debugging), or a connection via the Internet.



The sample programs described in this chapter may also be run with a RabbitSys-enabled RCM3365 operating in the RabbitSys mode. There is no change to the instructions when you use the serial programming cable. When you use an Ethernet cable, you may use CAT 5/6 straight-through Ethernet cables to connect the RCM3365 and your PC to a DHCP network. It is not necessary to use a crossover cable for a direct connection. Use the TCP/IP parameters such as the IP address that you identified with the rdiscover utility; if you are using an Ethernet crossover cable to connect the RCM3365 directly to your PC, use the TCP/IP parameters that you set up according to the instructions in Appendix E.

Table A-1 lists the electrical, mechanical, and environmental specifications for the RCM3365/RCM3375.

Table A-1. RabbitCore RCM3365/RCM3375 Specifications

Parameter	RCM3365	RCM3375
Microprocessor	Low-EMI Rabbit® 3000 at 44.2 MHz	
EMI Reduction	Spectrum spreader for reduced EMI (radiated emissions)	
Ethernet Port	10/100Base-T, RJ-45, 3 LEDs	
SRAM	512K program (fast SRAM) + 512K data	
Flash Memory (program)	512K	
Flash Memory (mass data storage)	32MB (fixed)* +*xD-Picture Card* with up to 128MB (NAND flash)	*xD-Picture Card* with up to 128MB (NAND flash)
LED Indicators	ACT (activity) LINK (link) SPEED (on for 100Base-T Ethernet connection) FM (flash memory, NAND) USR (user-programmable)	
Backup Battery	Connection for user-supplied backup battery (to support RTC and data SRAM)	
General-Purpose I/O	52 parallel digital I/0 lines: • 44 configurable I/O • 4 fixed inputs • 4 fixed outputs	
Additional Inputs	Startup mode (2), reset in	
Additional Outputs	Status, reset out	
External I/O Bus	Can be configured for 8 data lines and 5 address lines (shared with parallel I/O lines), plus I/O read/write	
Serial Ports	Six 3.3 V, CMOS-compatible ports (shared with I/O) • all 6 configurable as asynchronous (with IrDA) • 4 configurable as clocked serial (SPI) • 2 configurable as SDLC/HDLC • 1 asynchronous serial port dedicated for programming	
Serial Rate	Maximum asynchronous baud rate = CLK/8	
Slave Interface	A slave port allows the RCM3365/RCM3375 to be used as an intelligent peripheral device slaved to a master processor, which may either be another Rabbit 3000 or any other type of processor	
Real-Time Clock	Yes	
Timers	Ten 8-bit timers (6 cascadable, 3 reserved for internal peripherals), one 10-bit timer with 2 match registers	

Table A-1. RabbitCore RCM3365/RCM3375 Specifications (continued)

Parameter	RCM3365	RCM3375
Watchdog/ Supervisor	Yes	
Pulse-Width Modulators	4 PWM registers with 10-bit free-running counter and priority interrupts	
Input Capture	2-channel input capture can be used to time input signals from various port pins	
Quadrature Decoder	2-channel quadrature decoder accepts inputs from external incremental encoder modules	
Power	3.15–3.45 V DC 250 mA @ 44.2 MHz, 3.3 V	
Operating Temperature	-40°C to +70°C (boards manufactured up to May, 2008) 0°C to +70°C (boards manufactured after May, 2008)	
Humidity	5% to 95%, noncondensing	
Connectors	Two 2 × 17, 2 mm pitch one 2 × 5 for programming with 1.27 mm pitch one xD-Picture Card slot (RCM3365/RCM3375)	
Board Size	1.850" × 2.725" × 0.86" (47 mm × 69 mm × 22 mm)	

* RCM3365 modules sold before 2008 had 16MB fixed NAND flash memory.

NOTE: M-type *xD-Picture Cards* are ***not*** supported at this time.

B.6 Use of Rabbit 3000 Parallel Ports

Table B-5 lists the Rabbit 3000 parallel ports and their use for the Prototyping Board.

Table B-5. Prototyping Board Use of Rabbit 3000 Parallel Ports

Port	I/O	Use		Initial State
PA0–PA3	Data Bus	LCD/keypad module, motor driver, LEDs		Active high
PA4	Data Bus	LCD/keypad module, motor driver, relay and relay LED		Active high
PA5–PA7	Data Bus	LCD/keypad module, motor control		Active high
PB0	Input	CLKB, *xD-Picture Card* Detect		High
PB1	Input	CLKA Programming Port		High (when not driven by CLKA)
PB2–PB5	Address Bus	LCD/keypad module		High
PB6–PB7	Address Bus	—		High
PC0	Output	TXD SPI, serial flash	Serial Port D	High (SPI disabled)
PC1	Input	RXD SPI, serial flash		High (SPI disabled)
PC2	Output	TXC RS-485	Serial Port C	High (RS-485 disabled)
PC3	Input	RXC RS-485		High (RS-485 disabled)
PC4*	Output	TXB	Serial Port B	High (disabled)
PC5*	Input	RXB		High (disabled)
PC6	Output	TXA Programming Port	Serial Port A	High
PC7	Input	RXA Programming Port		High
PD0	Output	RCM3365/RCM3375 **USR** LED off (shared with NAND flash busy)		High
PD1	Output	Soldered-in NAND flash chip enable		High (disabled)
PD2	Output	SPI, serial flash		Low (SPI disabled)
PD3	Output	SPI, serial flash		High (SPI CS disabled)
PD4–PD6	Input	Serial flash		High (disabled)
PD7	Output	RS-485 Tx enable		Low (RS-485 Tx disabled)
PE0–PE1	Input	IN0–IN1		High
PE2	Output	Ethernet AEN, NAND flash function enable		High (disabled)
PE3	Output	Motor driver A clock pulse		Low (disabled)
PE4–PE5	Input	IN2–IN3, J8		High
PE6	Output	LCD/keypad module		High (disabled)

C.5 Mounting LCD/Keypad Module on the Prototyping Board

Install the LCD/keypad module on header sockets LCD1JA, LCD1JB, and LCD1JC of the Prototyping Board as shown in Figure C-7. Be careful to align the pins over the headers, and do not bend them as you press down to mate the LCD/keypad module with the Prototyping Board.

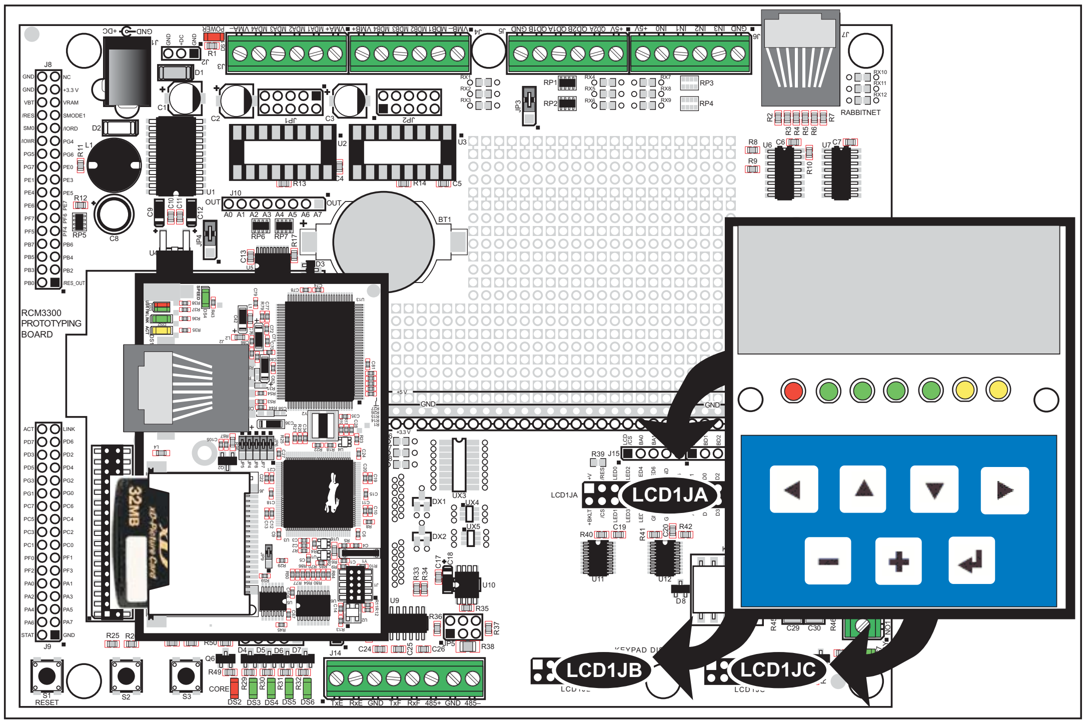


Figure C-7. Install LCD/Keypad Module on Prototyping Board

C.7 Sample Programs

Sample programs illustrating the use of the LCD/keypad module with the Prototyping Board are provided in the **`SAMPLES\RCM3360\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 RCM3365/RCM3375 must be connected to a PC using the serial programming cable (you also have the option to use an Ethernet cable if the RCM3365 is RabbitSys-enabled) 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. The red LED (DS3) on the RCM3365 module 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 glFillCircle(int xc, int yc, int rad);`

Draws a filled circle in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the circle that is outside the LCD display area will be clipped.

PARAMETERS

`xc` is the *x* coordinate of the center of the circle.

`yc` is the *y* coordinate of the center of the circle.

`rad` is the radius of the center of the circle (in pixels).

RETURN VALUE

None.

SEE ALSO

`glPlotCircle, glPlotPolygon, glFillPolygon`

`void glXFontInit(fontInfo *pInfo, char pixWidth, char pixHeight, unsigned startChar, unsigned endChar, unsigned long xmemBuffer);`

Initializes the font descriptor structure, where the font is stored in **`xmem`**.

PARAMETERS

`*pInfo` is a pointer to the font descriptor to be initialized.

`pixWidth` is the width (in pixels) of each font item.

`pixHeight` is the height (in pixels) of each font item.

`startChar` is the value of the first printable character in the font character set.

`endChar` is the value of the last printable character in the font character set.

`xmemBuffer` is the **`xmem`** pointer to a linear array of font bitmaps.

RETURN VALUE

None.

SEE ALSO

`glPrinf`

`void glPlotLine(int x0, int y0, int x1, int y1);`

Draws a line in the LCD buffer, and on the LCD if the buffer is unlocked. Any portion of the line that is beyond the LCD display area will be clipped.

PARAMETERS

`x0` is the *x* coordinate of one endpoint of the line.

`y0` is the *y* coordinate of one endpoint of the line.

`x1` is the *x* coordinate of the other endpoint of the line.

`y1` is the *y* coordinate of the other endpoint of the line.

RETURN VALUE

None.

SEE ALSO

`glPlotDot, glPlotPolygon, glPlotCircle`

`void glLeft1(int left, int top, int cols, int rows);`

Scrolls byte-aligned window left one pixel, right column is filled by current pixel type (color).

PARAMETERS

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

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

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

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

RETURN VALUE

None.

SEE ALSO

`glHScroll, glRight1`

`void glRight1(int left, int top, int cols, int rows);`

Scrolls byte-aligned window right one pixel, left column is filled by current pixel type (color).

PARAMETERS

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

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

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

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

RETURN VALUE

None.

SEE ALSO

`glHScroll, glLeft1`

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.

F.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.

`int rn_enable_wdt(int handle, int wdttype);`

Enables the hardware and/or software watchdog timers on a peripheral card. The software on the peripheral card will keep the hardware watchdog timer updated, but will hard reset if the time expires. The hardware watchdog cannot be disabled except by a hard reset on the peripheral card. The software watchdog timer must be updated by software on the master. The peripheral card will soft reset if the timeout set by **`rn_sw_wdt()`** expires. This function will check device information to determine that the peripheral card is connected to a master.

PARAMETERS

`handle` is an address index to device information. Use **`rn_device()`** or **`rn_find()`** to establish the handle.

`wdttype`

0 enables both hardware and software watchdog timers
1 enables hardware watchdog timer
2 enables software watchdog timer

RETURN VALUE

The status byte from the previous command. -1 means that device information indicates the peripheral card is not connected to the master.

SEE ALSO

`rn_hitwd, rn_sw_wdt`

`int rn_hitwd(int handle, char *count);`

Hits software watchdog. Set the timeout period and enable the software watchdog prior to using this function. This function will check device information to determine that the peripheral card is connected to a master.

PARAMETERS

`handle` is an address index to device information. Use **`rn_device()`** or **`rn_find()`** to establish the handle.

`count` is a pointer to return the present count of the software watchdog timer. The equivalent time left in seconds can be determined from **`count`** × 0.025 seconds.

RETURN VALUE

The status byte from the previous command. -1 means that device information indicates the peripheral card is not connected to the master.

SEE ALSO

`rn_enable_wdt, rn_sw_wdt`