Digi - 20-101-1055 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	Obsolete
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/20-101-1055

TABLE OF CONTENTS

Table 1 below summarizes the main features of the RCM3365 and the RCM3375 modules.

Table 1. RCM3365/RCM3375 Features

Feature	RCM3365	RCM3375
Microprocessor	Rabbit 3000 running at 44.2 MHz	
SRAM	512K program (fast SRAM) + 512K data	
Flash Memory (program)	512K	
Flash Memory (mass data storage)	32MB (fixed)* + up to 128MB (removable) (NAND flash)	up to 128MB (removable) (NAND flash)
Serial Ports	6 shared high-speed, 3.3 V CMOS-compatible ports: • all 6 are configurable as asynchronous serial ports; • 4 are configurable as a clocked serial port (SPI) and 1 is configurable as an HDLC serial port; • option for second HDLC serial port at the expense of 2 clocked serial ports (SPI)	

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

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

The RCM3365 and RCM3375 are programmed over a standard PC serial port through a serial programming cable supplied with the Development Kit, and can also be programed through a USB port with an RS-232/USB converter, or directly over an Ethernet link using the Dynamic C download manager with or without a RabbitLink; Dynamic C RabbitSys may also be used with a RabbitSys-enabled RCM3365 over an Ethernet link.

Appendix A provides detailed specifications for the RCM3365 and the RCM3375.

3.2.1 Use of NAND Flash

The following sample programs can be found in the **SAMPLES\RCM3360\NANDFlash** folder. As you run most of these sample programs, you will be prompted in the Dynamic C **STDIO** window to select either the soldered-in NAND flash (RCM3365 model only) or the socketed *xD-Picture Card* (0 = soldered, 1 = socketed).

- **NFLASH_DUMP.c**—This program is a utility for dumping the nonerased contents of a NAND flash chip to the Dynamic C **STDIO** window, and the contents may be redirected to a serial port.

  When the sample program starts running, it attempts to communicate with the user-selected NAND flash chip. If this communication is successful and the main page size is acceptable, the nonerased page contents (non 0xFF) from the NAND flash page are dumped to the Dynamic C **STDIO** win.for inspection.

  Note that an error message might appear when the first 32 pages (0x20 pages) are "dumped." You may ignore the error message.

- **NFLASH_INSPECT.c**—This program is a utility for inspecting the contents of a NAND flash chip. When the sample program starts running, it attempts to communicate with the NAND flash chip selected by the user. Once a NAND flash chip is found, the user can execute various commands to print out the contents of a specified page, clear (set to zero) all the bytes in a specified page, erase (set to FF), or write to specified pages.

  CAUTION: When you run this sample program, enabling the **#define NFLASH_CANERASEBADBLOCKS** macro makes it possible to write to bad blocks. The first two blocks on the *xD-Picture Card* are marked bad to protect the configuration data needed to use the card in a digital camera or a PC. You will only be able to use the *xD-Picture Card* in Rabbit-based systems if either of the first two blocks is written to.

- **NFLASH_LOG.c**—This program runs a simple Web server and stores a log of hits in the NAND flash. As long as the *xD-Picture Card* is plugged in to its connector J6, this sample program will log hits to the *xD-Picture Card*. Remove the *xD-Picture Card* if you wish to log hits on the soldered-in NAND flash (RCM3365 model only).

  This log can be viewed and cleared from a browser by connecting the RJ-45 jack on the RCM3365 to your PC as described in Section 6.1. The sidebar on the next page explains how to set up your PC or notebook to view this log.

3.2.5 RabbitNet

Sample programs are available for each RabbitNet peripheral card, and can be found in the Dynamic C **SAMPLES\RabbitNet** folder. When you run any of these sample programs in conjunction with the RCM3365/RCM3375 and the Prototyping Board, you need to add the line

```
#use rcm33xx.lib
```

at the beginning of the sample program.

TIP: You need to add **#use rcm33xx.lib** at the beginning of any sample program that is not in the Dynamic C **SAMPLES\RCM3360** folder.

3.2.6 Other Sample Programs

Section 6.6 describes the TCP/IP sample programs, and Appendix C.7 provides sample programs for the optional LCD/keypad module that can be installed on the Prototyping Board.

4.2 Serial Communication

The RCM3365/RCM3375 does not have any serial transceivers directly on the board. However, a serial interface may be incorporated into the board the RCM3365/RCM3375 is mounted on. For example, the Prototyping Board has RS-232 and RS-485 transceiver chips.

4.2.1 Serial Ports

There are six serial ports designated as Serial Ports A, B, C, D, E, and F. All six serial ports can operate in an asynchronous mode up to the baud rate of the system clock divided by 8. An asynchronous port can handle 7 or 8 data bits. A 9th bit address scheme, where an additional bit is sent to mark the first byte of a message, is also supported.

Serial Port A is normally used as a programming port, but may be used either as an asynchronous or as a clocked serial port once the RCM3365/RCM3375 has been programmed and is operating in the Run Mode.

Serial Port B is available on the RCM3365/RCM3375, and may be used as an asynchronous port. PB0 is used to enable Dynamic C to detect whether the *xD-Picture Card* is installed. If the card detect is not needed by your application program, you may remove R96 (see Figure A-5) to disable the *xD-Picture Card* detect, and then use Serial Port B as a clocked serial port.

Serial Ports C and D can also be operated in the clocked serial mode. In this mode, a clock line synchronously clocks the data in or out. Either of the two communicating devices can supply the clock.

Serial Ports E and F can also be configured as HDLC serial ports. The IrDA protocol is also supported in SDLC format by these two ports.

4.2.2 Ethernet Port

Figure 8 shows the pinout for the RJ-45 Ethernet port (J2). Note that some Ethernet connectors are numbered in reverse to the order used here.

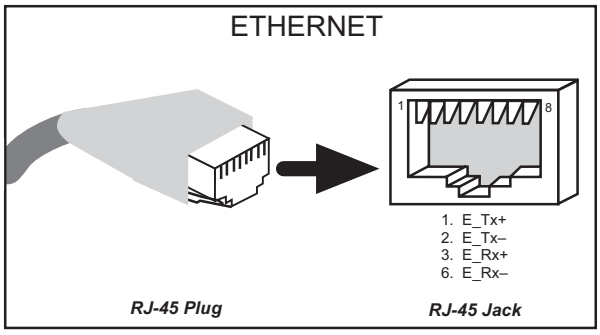


Figure 8. RJ-45 Ethernet Port Pinout

Three LEDs are placed next to the RJ-45 Ethernet jack, one to indicate an Ethernet link (**LINK**) one to indicate Ethernet activity (**ACT**), and one to indicate the 10/100Base-T speed.

The RJ-45 connector is shielded to minimize EMI effects to/from the Ethernet signals.

Figure 10 shows how to insert or remove the *xD-Picture Card*. While you remove or insert the *xD-Picture Card*, take care to avoid touching the electrical contacts on the bottom of the card to prevent electrostatic discharge damage to the card and to keep any moisture or other contaminants off the contacts. Do ***not*** remove or insert the *xD-Picture Card* while it is being accessed.

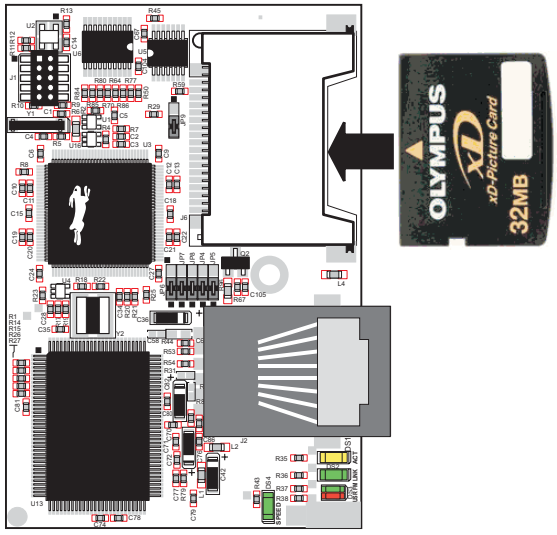


Figure 10. Insertion/Removal of xD-Picture Card

It is possible to hot-swap *xD-Picture Cards* without removing power from the RCM3365/RCM3375 module. The file system must be closed before the cards can be hot-swapped. The chip selects associated with the NAND flash and the *xD-Picture Card* must be set to their inactive state, and read/write operations addressed to the NAND flash area cannot be allowed to occur. These operations can be initiated in software by sensing an external switch actuated by the user, and the *xD-Picture Card* can then be removed and replaced with a different one. Once the application program detects a new card, the file system can be opened. These steps allow the *xD-Picture Card* to be installed or removed without affecting either the program, which continues to run on the RCM3365/RCM3375 module, or the data stored on the *xD-Picture Card*.

The **FAT_HOT_SWAP_336x0.C** sample program in the **SAMPLES\FileSystem** folder illustrates this hot-swapping procedure.

Rabbit recommends that you use header J6 only for the *xD-Picture Card* since other devices are not supported. Be careful to remove and insert the *xD-Picture Card* as shown, and be careful ***not*** to insert any foreign objects, which may short out the contacts and lead to the destruction of your *xD-Picture Card*.

Sample programs in the **SAMPLES\RCM3360\NANDFlash** folder illustrate the use of the NAND flash. These sample programs are described in Section 3.2.1, “Use of NAND Flash.” Pay careful attention to the sample programs to see how to close files and secure any data on the *xD-Picture Card* before you remove it.

void relayOut(int relay, int value);

Sets the position for the relay common contact. The default position is for normally closed contacts.

PARAMETERS

relay is the one relay (1)

value is the value used to connect the relay common contact:

0 = normally closed positions (NC1 and NC2)
1 = normally open positions (NO1 and NO2)

RETURN VALUE

None.

SEE ALSO

brdInit

5.2.6.4 Serial Communication

void ser485Tx(void);

Enables the RS-485 transmitter. Transmitted data are echoed back into the receive data buffer. The echoed data may be used as an indicator for disabling the transmitter by using one of the following methods:

Byte mode—disable the transmitter after the same byte that is transmitted is detected in the receive data buffer.

Block data mode—disable the transmitter after the same number of bytes transmitted are detected in the receive data buffer.

Remember to call the **serXopen()** function before running this function.

SEE ALSO

ser485Rx

void ser485Rx(void);

Disables the RS-485 transmitter. This puts the device into the listen mode, which allows it to receive data from the RS-485 interface.

Remember to call the **serXopen()** function before running this function.

SEE ALSO

ser485Tx

A.1 Electrical and Mechanical Characteristics

Figure A-1 shows the mechanical dimensions for the RCM3365/RCM3375.

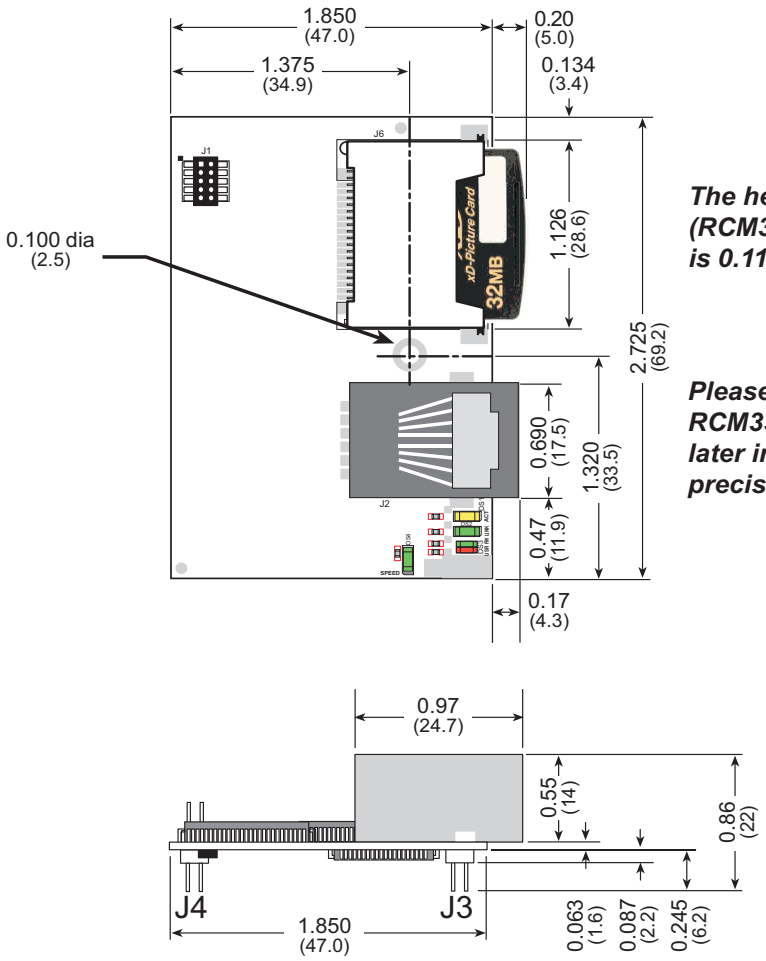


Figure A-1. RCM3365/RCM3375 Dimensions

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 RCM3365/RCM3375 in all directions when the RCM3365/RCM3375 is incorporated into an assembly that includes other printed circuit boards. An “exclusion zone” of 0.08" (2 mm) is recommended below the RCM3365/RCM3375 when the RCM3365/RCM3375 is plugged into another assembly. Figure A-2 shows this “exclusion zone.”

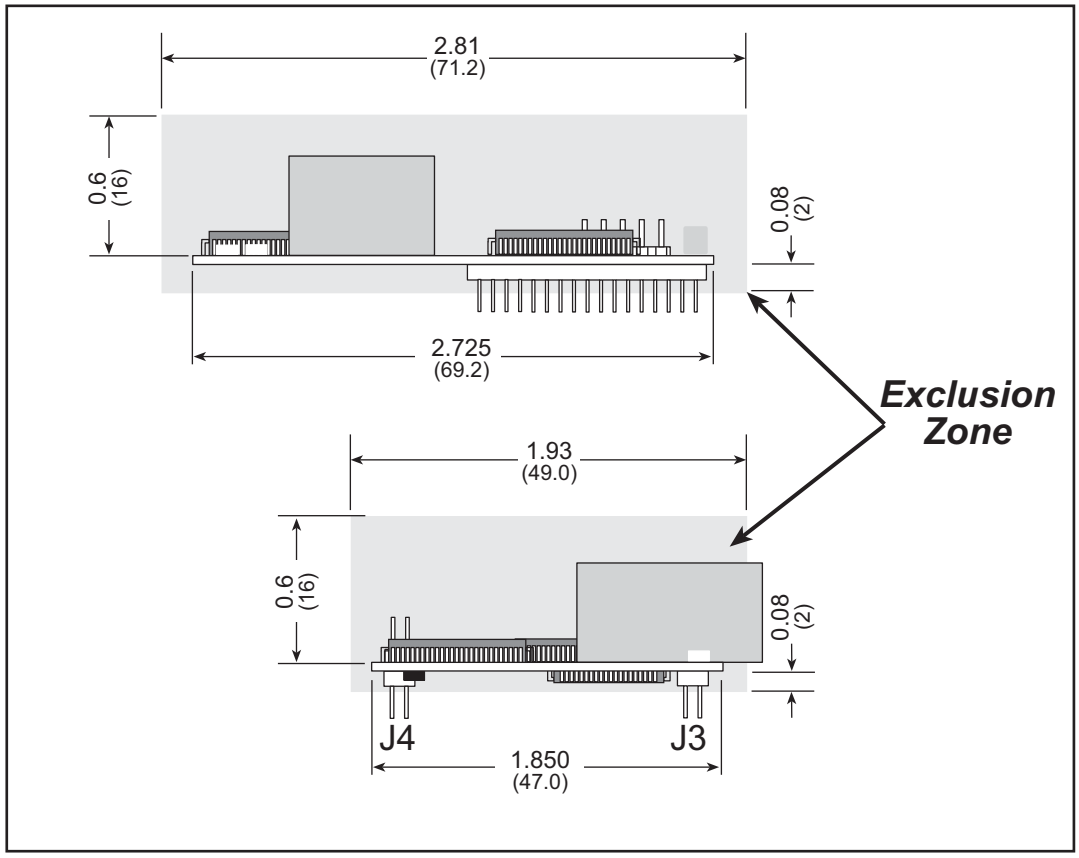


Figure A-2. RCM3365/RCM3375 “Exclusion Zone”

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

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.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.

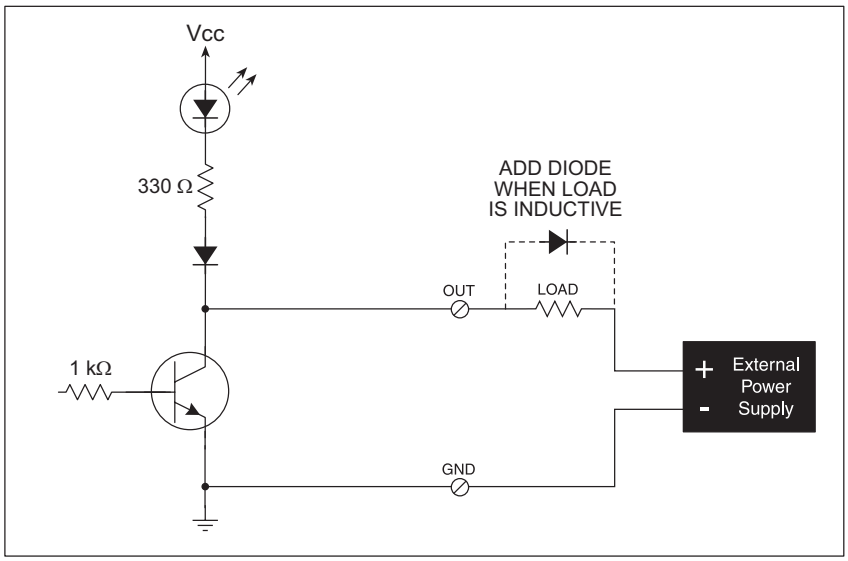


Figure B-6. Prototyping Board Sinking Digital Outputs

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.

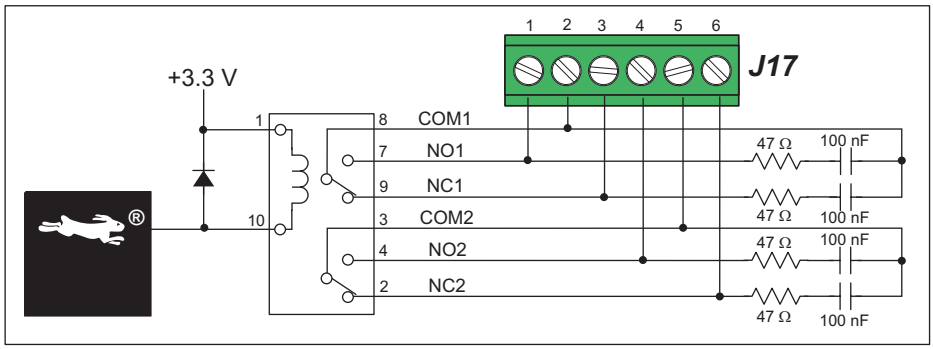


Figure B-7. Prototyping Board Relay Output Contact Connections

The relay is driven by pin PA4 of the RCM3365/RCM3375 module via U8, and is controlled by PE7 and PG5 as shown in the sample applications.

APPENDIX C. LCD/KEYPAD MODULE

An optional LCD/keypad is available for the Prototyping Board. Appendix C describes the LCD/keypad and provides the software function calls to make full use of the LCD/keypad.

C.1 Specifications

Two optional LCD/keypad modules—with or without a panel-mounted NEMA 4 water-resistant bezel—are available for use with the Prototyping Board. They are shown in Figure C-1.

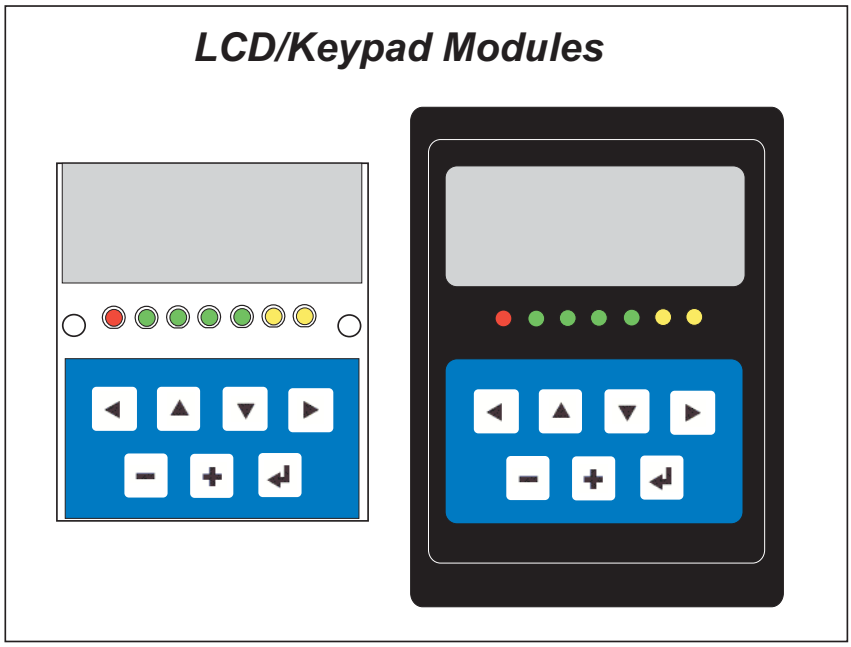


Figure C-1. LCD/Keypad Modules Versions

Only the version without the bezel can mount directly on the Prototyping Board; if you have the version with a bezel, you will have to remove the bezel to be able to mount the LCD/keypad module on the Prototyping Board. Either version of the LCD/keypad module can be installed at a remote location up to 60 cm (24") away. Contact your Rabbit sales representative or your authorized distributor for further assistance in purchasing an LCD/keypad module.

void glSetContrast(unsigned level);

Sets display contrast.

NOTE: This function is not used with the LCD/keypad module since the support circuits are not available on the LCD/keypad module.

void glFillScreen(char pattern);

Fills the LCD display screen with a pattern.

PARAMETER

The screen will be set to all black if **pattern** is 0xFF, all white if **pattern** is 0x00, and vertical stripes for any other pattern.

RETURN VALUE

None.

SEE ALSO

glBlock, glBlankScreen, glPlotPolygon, glPlotCircle

void glBlankScreen(void);

Blanks the LCD display screen (sets LCD display screen to white).

RETURN VALUE

None.

SEE ALSO

glFillScreen, glBlock, glPlotPolygon, glPlotCircle

void glBlock(int x, int y, int bmWidth, int bmHeight);

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

PARAMETERS

x is the *x* coordinate of the top left corner of the block.

y is the *y* coordinate of the top left corner of the block.

bmWidth is the width of the block.

bmWidth is the height of the block.

RETURN VALUE

None.

SEE ALSO

glFillScreen, glBlankScreen, glPlotPolygon, glPlotCircle

void glSwap(void);

Checks the LCD screen locking counter. The contents of the LCD buffer are transferred to the LCD if the counter is zero.

RETURN VALUE

None.

SEE ALSO

glBuffUnlock, glBuffLock, _glSwapData (located in the library specifically for the LCD that you are using)

void glSetBrushType(int type);

Sets the drawing method (or color) of pixels drawn by subsequent graphic calls.

PARAMETER

type value can be one of the following macros.

PIXBLACK draws black pixels (turns pixel on).
PIXWHITE draws white pixels (turns pixel off).
PIXXOR draws old pixel XOR'ed with the new pixel.

RETURN VALUE

None.

SEE ALSO

glGetBrushType

int glGetBrushType(void);

Gets the current method (or color) of pixels drawn by subsequent graphic calls.

RETURN VALUE

The current brush type.

SEE ALSO

glSetBrushType

void glPlotDot(int x, int y);

Draws a single pixel in the LCD buffer, and on the LCD if the buffer is unlocked. If the coordinates are outside the LCD display area, the dot will not be plotted.

PARAMETERS

x is the *x* coordinate of the dot.

y is the *y* coordinate of the dot.

RETURN VALUE

None.

SEE ALSO

glPlotline, glPlotPolygon, glPlotCircle

void glVScroll(int left, int top, int cols, int rows, int nPix);

Scrolls up or down, 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 up).

RETURN VALUE

None.

SEE ALSO

glHScroll

void glXPutBitmap(int left, int top, int width, int height, unsigned long bitmap);

Draws bitmap in the specified space. The data for the bitmap are stored in **xmem**. This function calls **glXPutFastmap** automatically if the bitmap is byte-aligned (the left edge and the width are each evenly divisible by 8).

Any portion of a bitmap image or character that is outside the LCD display area will be clipped.

PARAMETERS

left is the top left corner of the bitmap.

top is the top left corner of the bitmap.

width is the width of the bitmap.

height is the height of the bitmap.

bitmap is the address of the bitmap in **xmem**.

RETURN VALUE

None.

SEE ALSO

glXPutFastmap, glPrintf

void keypadDef();

Configures the physical layout of the keypad with the default ASCII return key codes.

Keypad physical mapping 1 x 7

0	4	1	5	2	6	3
['L']		['U']		['D']		['R']
	['–']		['+']		['E']	

where

'D' represents Down Scroll
'U' represents Up Scroll
'R' represents Right Scroll
'L' represents Left Scroll
'–' represents Page Down
'+' represents Page Up
'E' represents the ENTER key

Example: Do the followingfor the above physical vs. ASCII return key codes.

```
keyConfig ( 3,'R',0, 0, 0, 0, 0 );
keyConfig ( 6,'E',0, 0, 0, 0, 0 );
keyConfig ( 2,'D',0, 0, 0, 0, 0 );
keyConfig ( 4,'-',0, 0, 0, 0, 0 );
keyConfig ( 1,'U',0, 0, 0, 0, 0 );
keyConfig ( 5,'+',0, 0, 0, 0, 0 );
keyConfig ( 0,'L',0, 0, 0, 0, 0 );
```

Characters are returned upon keypress with no repeat.

RETURN VALUE

None.

SEE ALSO

keyConfig, keyGet, keyProcess

void keyScan(char *pcKeys);

Writes "1" to each row and reads the value. The position of a keypress is indicated by a zero value in a bit position.

PARAMETER

***pcKeys** is a pointer to the address of the value read.

RETURN VALUE

None.

SEE ALSO

keyConfig, keyGet, keypadDef, keyProcess

E.1 Load TCP/IP Parameters to the RCM3365 Module

1. Connect the 10-pin **PROG** connector of the serial programming cable to header J1 on the RCM3365 module as described in Section 2.2.2. (Do not use the **DIAG** connector.)
2. Use the **File** menu to open the sample program **SETUPFORCROSSOVER.C**, which is in the Dynamic C **SAMPLES\RABBITSYS** folder. Press function key **F9** to compile and run the program.

This sample program brings down the Ethernet interface, turns off DHCP, sets the RCM3365's IP address to 10.10.6.100, sets the netmask to 255.255.255.0, and sets the default gateway to 10.10.6.1. The RCM3365 module is now set up.

The IP and gateway addresses can be changed in the two macros in the two macros at the beginning of the sample program:

```
#define _IPADDR"10.10.6.100"
#define _GATEWAY"10.10.6.1"
```

To restore the RCM3365 to its default DHCP behavior, uncomment the **ENABLE_DHCP** macro:

```
//#define ENABLE_DHCP
```

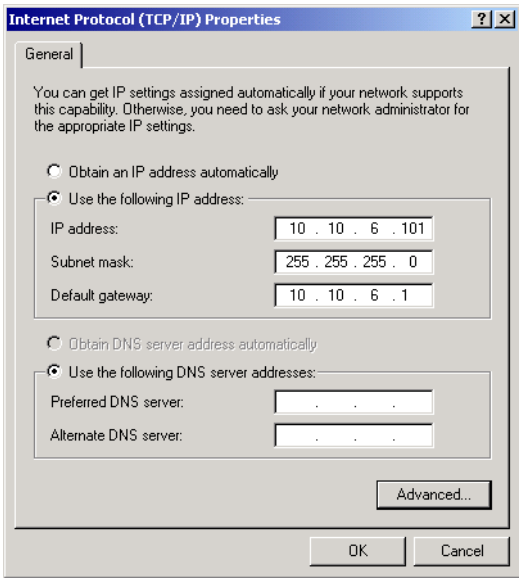
3. Now select the **IP Address** tab, and check **Specify an IP Address**, or select TCP/IP and click on “Properties” to fill in the following fields:

   IP Address : 10.10.6.101

   Netmask : 255.255.255.0

   Default gateway : 10.10.6.1

   TIP: If you are using a PC that is normally on a network, you will have disconnected the PC from that network. Write down the existing settings before changing them to facilitate restoring them when you are finished with the sample programs and reconnect your PC to the network.



4. Click **<OK>** or **<Close>** to exit the various dialog boxes.

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_rst_status(int handle, char *retdata);

Reads the status of which reset occurred and whether any watchdogs are enabled.

PARAMETERS

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

retdata is a pointer to the return address of the communication byte. A set bit indicates which error occurred. This register is cleared when read.

7—HW reset has occurred
6—SW reset has occurred
5—HW watchdog enabled
4—SW watchdog enabled
3,2,1,0—Reserved

RETURN VALUE

The status byte from the previous command.

int rn_comm_status(int handle, char *retdata);

PARAMETERS

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

retdata is a pointer to the return address of the communication byte. A set bit indicates which error occurred. This register is cleared when read.

7—Data available and waiting to be processed MOSI (master out, slave in)
6—Write collision MISO (master in, slave out)
5—Overrun MOSI (master out, slave in)
4—Mode fault, device detected hardware fault
3—Data compare error detected by device
2,1,0—Reserved

RETURN VALUE

The status byte from the previous command.