Digi - 20-101-1068 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), 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/20-101-1068

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.

2.3 Starting Dynamic C

NOTE: Dynamic C v. 9.60 or a later version is required if you are using an RCM3309 or an RCM3319 RabbitCore module.

Once the RCM3305 series module is connected as described in the preceding pages, start Dynamic C by double-clicking on the Dynamic C icon on your desktop or in your **Start** menu. Select **Code and BIOS in Flash, Run in RAM** on the "Compiler" tab in the Dynamic C **Options > Project Options** menu. Click **OK**.

If you are using a USB port to connect your computer to the RCM3305/RCM3315 module, choose **Options > Project Options** and select "Use USB to Serial Converter" on the **Communications** tab. Click **OK**.

2.4 Run a Sample Program

Use the **File** menu to open the sample program **PONG.C**, which is in the Dynamic C **SAMPLES** folder. Press function key **F9** to compile and run the program. The **STDIO** window will open on your PC and will display a small square bouncing around in a box.

This program shows that the CPU is working. The sample program described in Section 6.5, "Run the PINGME.C Sample Program," tests the TCP/IP portion of the board.

2.4.1 Troubleshooting

If Dynamic C cannot find the target system (error message **"No Rabbit Processor Detected."**):

- Check that the RCM3305 series module is powered correctly — the red **CORE** LED on the Prototyping Board should be lit when the module is mounted on the Prototyping Board and the AC adapter is plugged in.
- Check both ends of the programming cable to ensure that they are firmly plugged into the PC and the **PROG** connector, not the **DIAG** connector, is plugged in to the programming port on the RCM3305 series module with the marked (colored) edge of the programming cable towards pin 1 of the programming header.
- Ensure that the RCM3305 series module is firmly and correctly installed in its connectors on the Prototyping Board.
- Dynamic C uses the COM port or USB port specified during installation. Select a different COM port within Dynamic C. From the **Options** menu, select **Project Options**, then select **Communications**. Select another COM port from the list, then click **OK**. Press **<Ctrl-Y>** to force Dynamic C to recompile the BIOS. If Dynamic C still reports it is unable to locate the target system, repeat the above steps until you locate the COM port used by the programming cable.
- If you get an error message when you plugged the programming cable into a USB port, you will have to install USB drivers. Drivers for Windows XP are available in the Dynamic C **Drivers\Rabbit USB Programming Cable\WinXP_2K** folder — double-click **DPInst.exe** to install the USB drivers. Drivers for other operating systems are available online at www.ftdichip.com/Drivers/VCP.htm.

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.

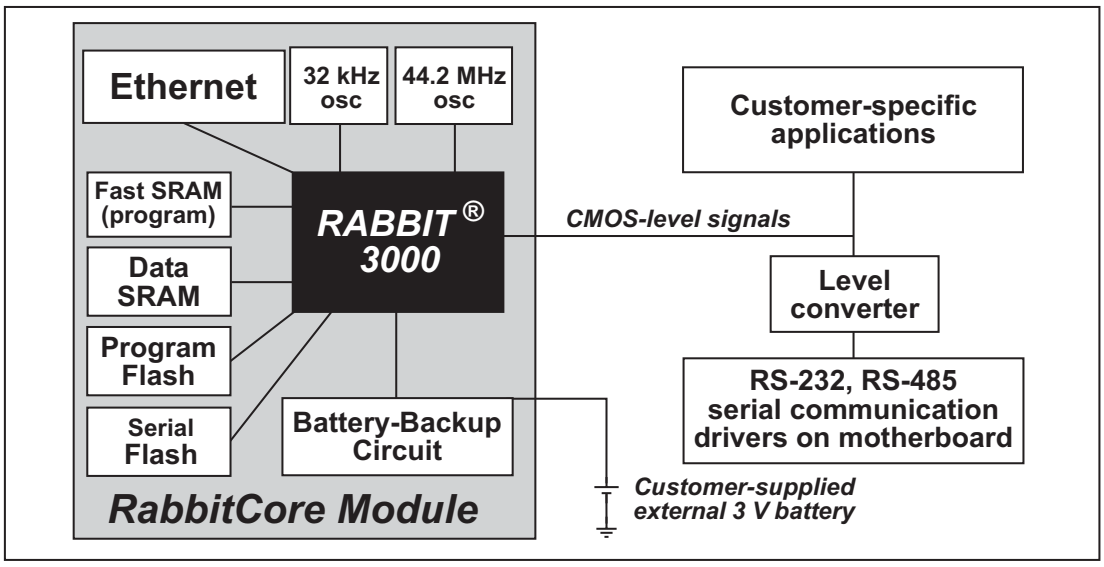

Figure 4. RCM3305/RCM3315 Subsystems

4.2.3 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.

4.4 Other Hardware

4.4.1 Clock Doubler

The RCM3305/RCM3315 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 RCM3305/RCM3315 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.

4.4.2 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 unnecessary for the RCM3305/RCM3315.
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.

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.

APPENDIX A. RCM3305/RCM3315 SPECIFICATIONS

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

Table A-1. RCM3305/RCM3315 Specifications (continued)

Parameter	RCM3305	RCM3315
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	
Board Size	1.850" × 2.725" × 0.86" (47 mm × 69 mm × 22 mm)	

Table A-8 lists the configuration options.

Table A-8. RCM3305/RCM3315 Jumper Configurations

Header	Description	Pins Connected		Factory Default
JP1	Flash Memory Size	1–2	128K/256K	
		2–3	512K	✕
JP2	Flash Memory Bank Select	1–2	Reserved for future use	
		2–3	Normal Mode	✕
JP3	Data SRAM Size	1–2	128K/256K	
		2–3	512K	✕
JP4	Ethernet or I/O Output on Header J3	1–2	TPO+	
		2–3	PD3	✕
JP5	Ethernet or I/O Output on Header J3	1–2	TPO–	
		2–3	PD2	✕
JP6	Ethernet or I/O Output on Header J3	1–2	ENET_INT	
		2–3	PE0	✕
JP7	Ethernet or I/O Output on Header J3	1–2	TPI+	
		2–3	PD7	✕
JP8	Ethernet or I/O Output on Header J3	1–2	TPI–	
		2–3	PD6	✕

NOTE: The jumper connections are made using 0 Ω surface-mounted resistors.

APPENDIX B. PROTOTYPING BOARD

Appendix B describes the features and accessories of the Prototyping Board.

B.1 Introduction

The Prototyping Board included in the Development Kit makes it easy to connect an RCM3305/RCM3315 module to a power supply and a PC workstation for development. It also provides some basic I/O peripherals (RS-232, RS-485, a relay, LEDs, and switches), as well as a prototyping area for more advanced hardware development.

For the most basic level of evaluation and development, the Prototyping Board can be used without modification.

As you progress to more sophisticated experimentation and hardware development, modifications and additions can be made to the board without modifying or damaging the RCM3305/RCM3315 module itself.

The Prototyping Board is shown below in Figure B-1, with its main features identified.

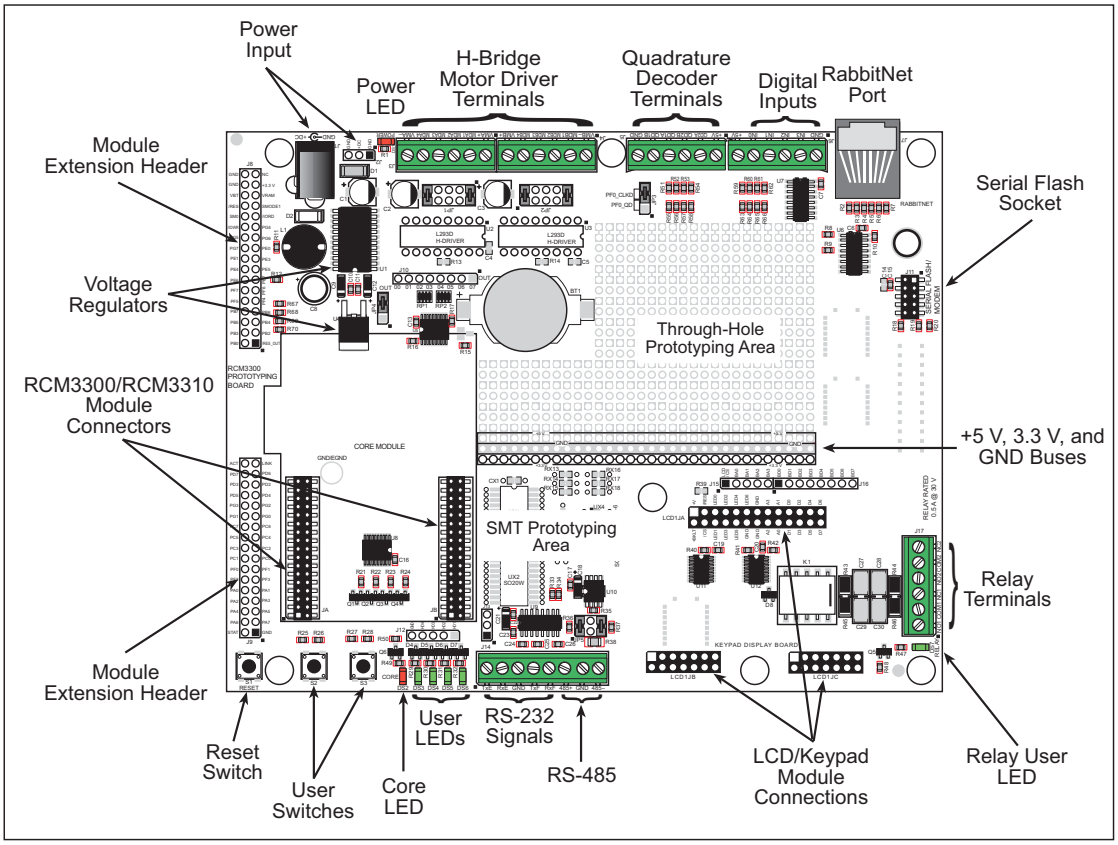

Figure B-1. Prototyping Board

- **Module Extension Headers**—The complete pin set of the RCM3305/RCM3315 module is duplicated at headers J8 and J9. Developers can solder wires directly into the appropriate holes, or, for more flexible development, 2 × 17 header strips with a 0.1" pitch can be soldered into place. See Figure B-4 for the header pinouts.
- **Digital I/O**—Four digital inputs are available on screw-terminal header J6. See Figure B-4 for the header pinouts.
- **RS-232**—Two 3-wire serial ports or one 5-wire RS-232 serial port are available on the Prototyping Board at screw-terminal header J14.
- **RS-485**—One RS-485 serial port is available on the Prototyping Board at screw-terminal header J14.
- **Quadrature Decoder**—Four quadrature decoder inputs (PF0–PF3) from the Rabbit 3000 chip are available on screw-terminal header J5. See Figure B-4 for the header pinouts.
- **H-Bridge Motor Driver**—Two pairs of H-bridge motor drivers are supported using screw-terminal headers J3 and J4 on the Prototyping Board for stepper-motor control. See Figure B-4 for the header pinouts.
- **RabbitNet Port**—One RS-422 RabbitNet port (shared with the serial flash interface) is available to allow RabbitNet peripheral cards to be used with the Prototyping Board.
- **Serial Flash Interface**—One serial flash interface (shared with the RabbitNet port) is available to allow Rabbit's SF1000 series serial flash to be used on the Prototyping Board.

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

Table B-1. Prototyping Board Specifications

Parameter	Specification
Board Size	5.25" × 6.75" × 1.00" (133 mm × 171 mm × 25 mm)
Operating Temperature	–20°C to +70°C
Humidity	5% to 95%, noncondensing
Input Voltage	8 V to 30 V DC
Maximum Current Draw (including user-added circuits)	800 mA max. for +3.3 V supply, 1 A total +3.3 V and +5 V combined
Backup Battery	CR2032, 3 V lithium coin-type
Digital Inputs	4 inputs pulled up, ± 36 V DC, switching threshold 0.9–2.3 V typical
Digital Outputs	4 sinking outputs,+30 V DC, 500 mA maximum per channel 8 CMOS-level outputs if stepper motor not installed
Relay	SPDT relay, 500 mA @ 30 V
Serial Ports	• two 3-wire RS-232 *or* one RS-232 with RTS/CTS • one RS-485
Other Serial Interfaces	RabbitNet RS-422 port *or* serial flash interface
Other Interfaces	• stepper motor control • quadrature decoder • LCD/keypad module
LEDs	Seven LEDs • one power on indicator • one RCM3305/RCM3315 module indicator • four user-configurable LEDs • one relay indicator
Prototyping Area	Throughhole, 0.1" spacing, additional space for SMT components
Connectors	• two 2 × 17, 2 mm pitch sockets for RCM3305/RCM3315 module • one 2 × 5, 2 mm pitch socket for serial flash • six screw-terminal headers for serial ports, digital inputs, stepper motor control, quadrature decoder, and relay contacts • one RJ-45 RabbitNet jack
Standoffs/Spacers	7, accept 4-40 x 1/2 screws

B.4.8 Other Prototyping Board Modules

An optional LCD/keypad module is available that can be mounted on the Prototyping Board. The signals on headers LCD1JB and LCD1JC will be available only if the LCD/keypad module is installed. Refer to Appendix C, "LCD/Keypad Module," for complete information.

Rabbit's SF1000 series serial flash may be installed in the socket labeled J11. The J11 interface is enabled in software by setting PD2 = 0. Header JP3 must have pins 2–3 jumpered when using the J11 interface. Note that the RabbitNet port and the J11 interface cannot be used simultaneously.

B.4.9 Quadrature Decoder

Four quadrature decoder inputs are available on screw-terminal header J5. To use the PF0 input from the Rabbit microprocessor, which goes to the QD1B input, remember to reconfigure the jumper on header JP3 to jumper pins 1–2.

Additional information on the use of the quadrature decoders on Parallel Port F is provided in the ***Rabbit 3000 Microprocessor User's Manual***.

B.4.10 Stepper-Motor Control

The Prototyping Board can be used to demonstrate the use of the RCM3305/RCM3315 to control a stepper motor. Stepper motor control typically directs moves in two orthogonal directions, and so two sets of stepper-motor control circuits are provided for via screw-terminal headers J3 and J4.

In order to use the stepper-motor control, install two Texas Instruments L293DN chips at locations U2 and U3 (shown in Figure B-10). These chips are readily available from your favorite electronics parts source, and may be purchased through Rabbit's Web store as part number 660-0205.

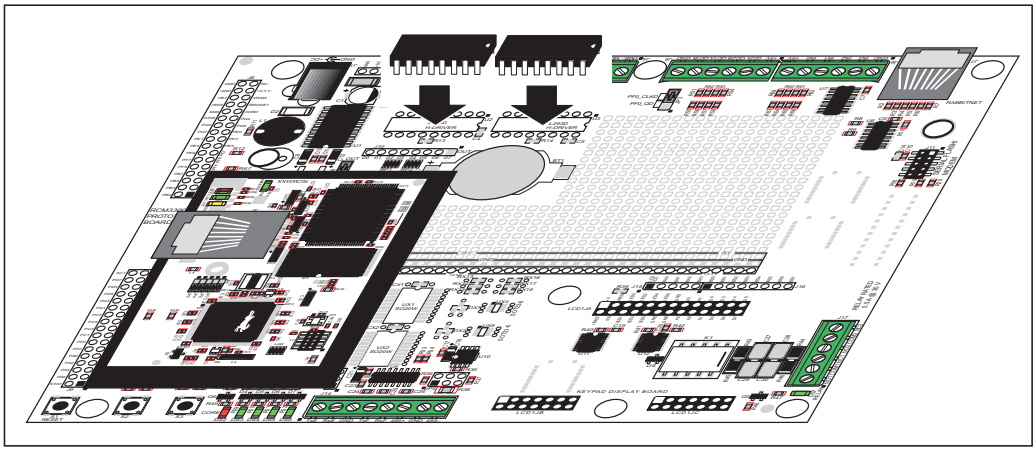

Figure B-10. Install Four-Channel Push-Pull Driver Chips

void glFillPolygon(int n, int x1, int y1, int x2, int y2, ...);

Fills a polygon in the LCD page buffer and on the LCD if the buffer is unlocked. Any portion of the polygon that is outside the LCD display area will be clipped. If fewer than 3 vertices are specified, the function will return without doing anything.

PARAMETERS

n is the number of vertices.

x1 is the *x* coordinate of the first vertex.

y1 is the *y* coordinate of the first vertex.

x2 is the *x* coordinate of the second vertex.

y2 is the *y* coordinate of the second vertex.

... are the coordinates of additional vertices.

RETURN VALUE

None.

SEE ALSO

glFillVPolygon, glPlotPolygon, glPlotVPolygon

void glPlotCircle(int xc, int yc, int rad);

Draws the outline of a 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

glFillCircle, glPlotPolygon, glFillPolygon

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 glPutChar(char ch, char *ptr, int *cnt, glPutCharInst *pInst)

Provides an interface between the **STDIO** string-handling functions and the graphic library. The **STDIO** string-formatting function will call this function, one character at a time, until the entire formatted string has been parsed. Any portion of the bitmap character that is outside the LCD display area will be clipped.

PARAMETERS

ch is the character to be displayed on the LCD.

***ptr** is not used, but is a place holder for **STDIO** string functions.

***cnt** is not used, is a place holder for **STDIO** string functions.

pInst is a pointer to the font descriptor.

RETURN VALUE

None.

SEE ALSO

glPrintf, glPutFont, doprnt

void glPrintf(int x, int y, fontInfo *pInfo, char *fmt, ...);

Prints a formatted string (much like **printf**) on the LCD screen. Only the character codes that exist in the font set are printed, all others are skipped. For example, '\b', '\t', '\n' and '\r' (ASCII backspace, tab, new line, and carriage return, respectively) will be printed if they exist in the font set, but will not have any effect as control characters. Any portion of the bitmap character that is outside the LCD display area will be clipped.

PARAMETERS

x is the *x* coordinate (column) of the upper left corner of the text.

y is the *y* coordinate (row) of the upper left corner of the text.

pInfo is a pointer to the font descriptor.

***fmt** is a formatted string.

... are formatted string conversion parameter(s).

EXAMPLE

```
glprintf(0,0, &fi12x16, "Test %d\n", count);
```

RETURN VALUE

None.

SEE ALSO

glXFontInit