PRODUCT MANUAL




# Intellicom

Models OP6600 and OP6700

## User's Manual

019–0078 • 090529–J

# Intellicom User's Manual

Part Number 019-0078 • 090529–J • Printed in U.S.A.

©2000–2009 Digi International Inc. • All rights reserved.

No part of the contents of this manual may be reproduced or transmitted in any form or by any means without the express written permission of Digi International.

Permission is granted to make one or more copies as long as the copyright page contained therein is included. These copies of the manuals may not be let or sold for any reason without the express written permission of Digi International.

Digi International reserves the right to make changes and improvements to its products without providing notice.

## Trademarks

Rabbit and Dynamic C are registered trademarks of Digi International Inc.

Rabbit 2000 and RabbitCore are trademarks of Digi International Inc.

The latest revision of this manual is available on the Rabbit Web site, www.rabbit.com, for free, unregistered download.

**Digi International Inc.**

www.rabbit.com

# TABLE OF CONTENTS


**Chapter 1. Introduction 1**

1.1 Features ..... 1
1.2 Development and Evaluation Tools ..... 3
1.2.1 Tool Kit ..... 3
1.2.2 Software ..... 3
1.3 CE Compliance ..... 4
1.3.1 Design Guidelines ..... 5
1.3.2 Interfacing the Intellicom to Other Devices ..... 5

**Chapter 2. Getting Started 7**

2.1 Power Supply Connections ..... 8
2.2 Demonstration Program on Power-Up ..... 10
2.3 Programming Cable Connections ..... 12
2.4 Installing Dynamic C ..... 13
2.5 Starting Dynamic C ..... 14
2.6 PONG.C ..... 15
2.7 Where Do I Go From Here? ..... 16

**Chapter 3. Subsystems 17**

3.1 Intellicom Subsystems ..... 18
3.1.1 Digital Inputs ..... 19
3.1.2 Digital Outputs ..... 19
3.2 Serial Communication ..... 20
3.2.1 RS-232 ..... 23
3.2.2 RS-485 ..... 23
3.2.3 Programming Port ..... 26
3.3 Programming Cable ..... 27
3.3.1 Changing Between Program Mode and Run Mode ..... 27
3.4 Memory ..... 28
3.4.1 SRAM ..... 28
3.4.2 Flash Memory ..... 28
3.4.3 Dynamic C BIOS Source Files ..... 28
3.5 Speaker ..... 29
3.6 Vacuum Fluorescent Display ..... 29
3.7 Other Hardware ..... 30
3.7.1 Clock Doubler ..... 30
3.7.2 Spectrum Spreader ..... 30

**Chapter 4. Software 31**

4.1 Running Dynamic C ..... 31
4.1.1 Upgrading Dynamic C ..... 33
4.1.1.1 Patches and Bug Fixes ..... 33
4.1.1.2 Upgrades ..... 33
4.2 Sample Programs ..... 34
4.2.1 General Intellicom Operation ..... 34
4.3 Dynamic C Libraries ..... 36

4.4 Intellicom Function Calls ..... 37
4.4.1 Board Initialization ..... 37
4.4.2 Digital I/O ..... 37
4.4.3 Serial Communication ..... 38
4.4.4 Keypad Controls ..... 39
4.4.5 Display Controls ..... 41
4.4.6 Speaker Controls ..... 44

**Chapter 5. Using the TCP/IP Features 45**
5.1 TCP/IP Connections ..... 45
5.2 Running TCP/IP Sample Programs ..... 47
5.3 How to Set IP Addresses in the Sample Programs ..... 49
5.4 How to Set Up Your Computer's IP Address For Direct Connect ..... 50
5.5 Run the PINGME.C Demo ..... 51
5.6 Running More Demo Programs With a Direct Connection ..... 51
5.7 Where Do I Go From Here? ..... 52

**Appendix A. Intellicom Specifications 53**
A.1 Electrical and Mechanical Specifications ..... 54
A.2 Conformal Coating ..... 56
A.3 Jumper Configurations ..... 57

**Appendix B. Keypad and Plastic Enclosure 59**
B.1 Keypad Insert ..... 60
B.2 Plastic Enclosure ..... 62
B.2.1 Assembling Intellicom Enclosure ..... 64
B.2.1.1 Custom Mounting In An Opening ..... 64
B.2.1.2 Supplied Outer Casing ..... 65

**Appendix C. Power Management 67**
C.1 Power Supplies ..... 67
C.2 Batteries and External Battery Connections ..... 68
C.2.1 Battery-Backup Circuit ..... 68
C.2.2 Power to VRAM Switch ..... 69
C.2.3 Reset Generator ..... 70
C.2.4 Replacing the Backup-Battery Board ..... 70
C.3 Chip Select Circuit ..... 71

**Appendix D. Running Sample Programs 73**
D.1 Connecting Demonstration Board ..... 74
D.2 Running Sample Program DEMOBRD1.C ..... 75
D.2.1 Single-Stepping ..... 76
D.2.1.1 Watch Expression ..... 76
D.2.1.2 Break Point ..... 76
D.2.1.3 Editing the Program ..... 77
D.2.1.4 Watching Variables Dynamically ..... 77
D.2.1.5 Summary of Features ..... 77
D.2.2 Cooperative Multitasking ..... 78
D.2.3 Advantages of Cooperative Multitasking ..... 80

**Index 81**

**Schematics 85**

# 1. INTRODUCTION

The Intellicom intelligent terminal interface is a high-performance, C-programmable terminal that offers built-in I/O and Ethernet connectivity. A Rabbit® 2000 microprocessor operating at 18.5 MHz provides fast data processing.

## 1.1 Features

- C-programmable to create a custom user interface
- 4 protected logic-level digital inputs
- 4 protected sinking digital outputs
- High-visibility backlit 4 × 20 LCD
- 10Base-T Ethernet interface
- TCP/IP capability
- RS-232 and RS-485 serial ports
- 128K SRAM and 256K–512K flash EPROM
- Self-healing lens is scratch, impact, and abrasion-resistant
- Real-time clock
- Watchdog supervisor
- Voltage regulator
- Backup battery
- Can be programmed to emulate a serial terminal
- Splash-resistant when panel-mounted using the supplied gasket
- Can be wall-mounted or panel-mounted

Appendix A provides detailed specifications for the Intellicom.

Two versions of the Intellicom are available. Their standard features are summarized in Table 1.

***Table 1. Intellicom Series Features***

| Model | Features |
|---|---|
| OP6600 | Standard terminal *without* Ethernet interface and only 256K flash EPROM. |
| OP6700 | Full-featured terminal *with* Ethernet interface and 512K flash EPROM. |

Both models are available with a vacuum fluorescent display instead of the LCD.

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

## 1.2 Development and Evaluation Tools

### 1.2.1 Tool Kit

The Tool Kit has the essentials that you need to understand and program your own Rabbit-based display unit.

The items in the Tool Kit and their use are as follows:

- ***Intellicom Getting Started*** instructions.
- ***Dynamic C*** CD-ROM, with complete product documentation on disk.
- Demonstration Board. The Demonstration Board includes pushbutton switches and LEDs, and can be connected to the Intellicom board. Programs that run on the Demonstration Board can be used to flash the LEDs and otherwise demonstrate the capabilities of the Intellicom terminal.
- Programming cable. The programming cable is used to connect your PC serial port to the Intellicom to write and debug C programs that run on the Intellicom board.
- Wire assembly to connect Intellicom board to Demonstration Board.
- Screwdriver.
- Universal AC adapter (includes Canada/Japan/U.S., Australia/N.Z., U.K., and European style plugs). The AC adapter is used to power the Intellicom board. A power supply of 12 V at up to 500 mA is recommended. The Intellicom can also be powered from any DC voltage source between 9.0 V and 40 V.

### 1.2.2 Software

The Intellicom is programmed using Rabbit's Dynamic C. A compatible version is included on the Tool Kit CD-ROM. Dynamic C v. 9.60 includes the popular µC/OS-II real-time operating system, point-to-point protocol (PPP), FAT file system, RabbitWeb, and other select libraries that were previously sold as individual Dynamic C modules.

Rabbit also offers for purchase the Rabbit Embedded Security Pack featuring the Secure Sockets Layer (SSL) and a specific Advanced Encryption Standard (AES) library. In addition to the Web-based technical support included at no extra charge, a one-year telephone-based technical support subscription is also available for purchase. Visit our Web site at www.rabbit.com for further information and complete documentation, or contact your Rabbit sales representative or authorized distributor.

## 1.3 CE Compliance

Equipment is generally divided into two classes.

| CLASS A | CLASS B |
|---|---|
| Digital equipment meant for light industrial use | Digital equipment meant for home use |
| Less restrictive emissions requirement:<br>less than 40 dB µV/m at 10 m<br>(40 dB relative to 1 µV/m) or 300 µV/m | More restrictive emissions requirement:<br>30 dB µV/m at 10 m or 100 µV/m |

These limits apply over the range of 30–230 MHz. The limits are 7 dB higher for frequencies above 230 MHz. Although the test range goes to 1 GHz, the emissions from Rabbit-based systems at frequencies above 300 MHz are generally well below background noise levels.

The OP6700 operator interface has been tested and was found to be in conformity with the following applicable immunity and emission standards. The OP6600 operator interface is also CE qualified as it is a subversion of the OP6700 operator interface. Boards that are CE-compliant have the CE mark.



**NOTE:** Earlier versions of the Intellicom sold before 2003 that do not have the CE mark are ***not*** CE-complaint.

### Immunity

The Intellicom series of operator interfaces meets the following EN55024/1998 immunity standards.

- EN61000-4-2 (ESD)
- EN61000-4-3 (Radiated Immunity)
- EN61000-4-4 (EFT)
- EN61000-4-6 (Conducted Immunity)

Additional shielding or filtering may be required for a heavy industrial environment.

### Emissions

The Intellicom series of operator interfaces meets the following emission standards emission standards with the Rabbit 2000 spectrum spreader turned on and set to the normal mode. The spectrum spreader is only available with Rev. C or higher of the Rabbit 2000 microprocessor. This microprocessor is used on the Intellicom operator interfaces that carry the CE mark.

- EN55022:1998 Class B
- FCC Part 15 Class B

Your results may vary, depending on your application, so additional shielding or filtering may be needed to maintain the Class B emission qualification.

### 1.3.1 Design Guidelines

Note the following requirements for incorporating the Intellicom series of operator interfaces into your application to comply with CE requirements.

#### General

- The power supply provided with the Tool Kit is for development purposes only. It is the customer's responsibility to provide a CE-compliant power supply for the end-product application.
- When connecting the Intellicom to outdoor cables, the customer is responsible for providing CE-approved surge/lighting protection.
- Rabbit recommends placing digital I/O or analog cables that are 3 m or longer in a metal conduit to assist in maintaining CE compliance and to conform to good cable design practices.
- When installing or servicing the Intellicom, it is the responsibility of the end-user to use proper ESD precautions to prevent ESD damage to the Intellicom.

#### Safety

- All inputs and outputs to and from the Intellicom must not be connected to voltages exceeding SELV levels (42.4 V AC peak, or 60 V DC).
- The lithium backup battery circuit on the Intellicom has been designed to protect the battery from hazardous conditions such as reverse charging and excessive current flows. Do not disable the safety features of the design.

### 1.3.2 Interfacing the Intellicom to Other Devices

Since the Intellicom operator interfaces are designed to be connected to other devices, good EMC practices should be followed to ensure compliance. CE compliance is ultimately the responsibility of the integrator. Additional information, tips, and technical assistance are available from your authorized Rabbit distributor, and are also available on our Web site at www.rabbit.com.

# 2. GETTING STARTED

Chapter 2 explains how to connect the power supply to the Intellicom board and how to connect the programming cable from the Intellicom board to your PC. Once you run a sample program to demonstrate that you have connected everything correctly, you will be ready to go on and finish developing your system.

## 2.1 Power Supply Connections

### 1. Remove and set aside outer casing and rubber gasket.

Before proceeding, remove and set aside the outer casing, rubber gasket, screws, and panel-mount brackets included with your Intellicom unit, shown in Figure 1. The outer casing and rubber gasket are not attached to the front panel when the Intellicom is shipped. Take care not to damage the rubber gasket.

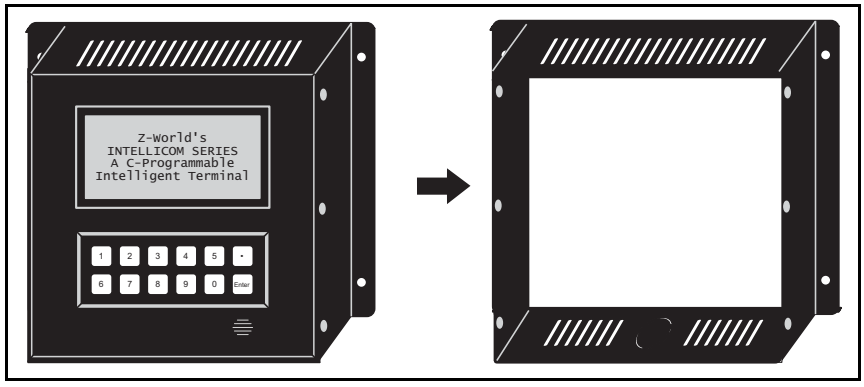



***Figure 1. Remove and Set Aside Outer Casing and Accessory Parts***

### 2. Position Intellicom board.

The Intellicom board is attached to the back of the front panel. Turn the front panel assembly over so that the Intellicom board is facing up as shown in Figure 2.

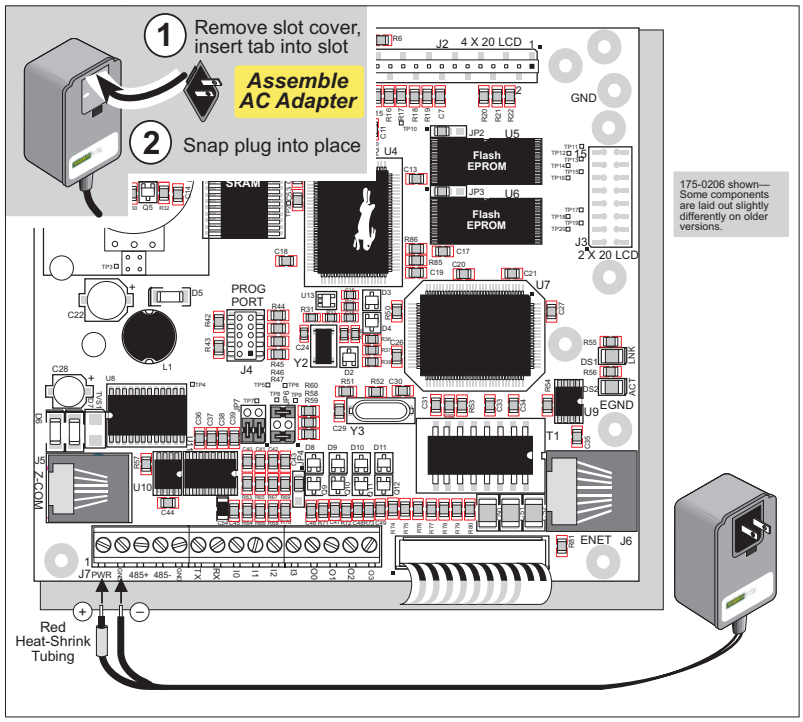



***Figure 2. Power Supply Connections***

### 3. Connect Power Supply to the Intellicom Board

First, prepare the AC adapter for the country where it will be used by selecting the plug. The Intellicom Tool 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 2, then press down on the spring-loaded clip below the plug assembly to allow the plug assembly to click into place.

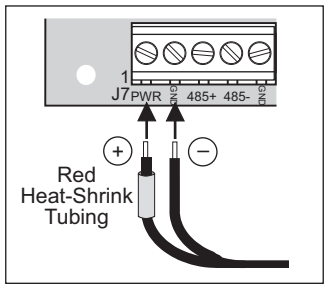



***Figure 3. Power Supply Connections—Detailed View***

Connect the positive lead (indicated with red heat-shrink tubing on the AC adapter included with the Tool Kit) to the PWR connector on header J7 on the Intellicom Board and connect the negative lead to GND on header J7 as shown in Figure 2 and Figure 3.

**CAUTION:** Be careful to hook up the positive and negative power leads ***exactly*** as described. Otherwise, the Intellicom board will not function.

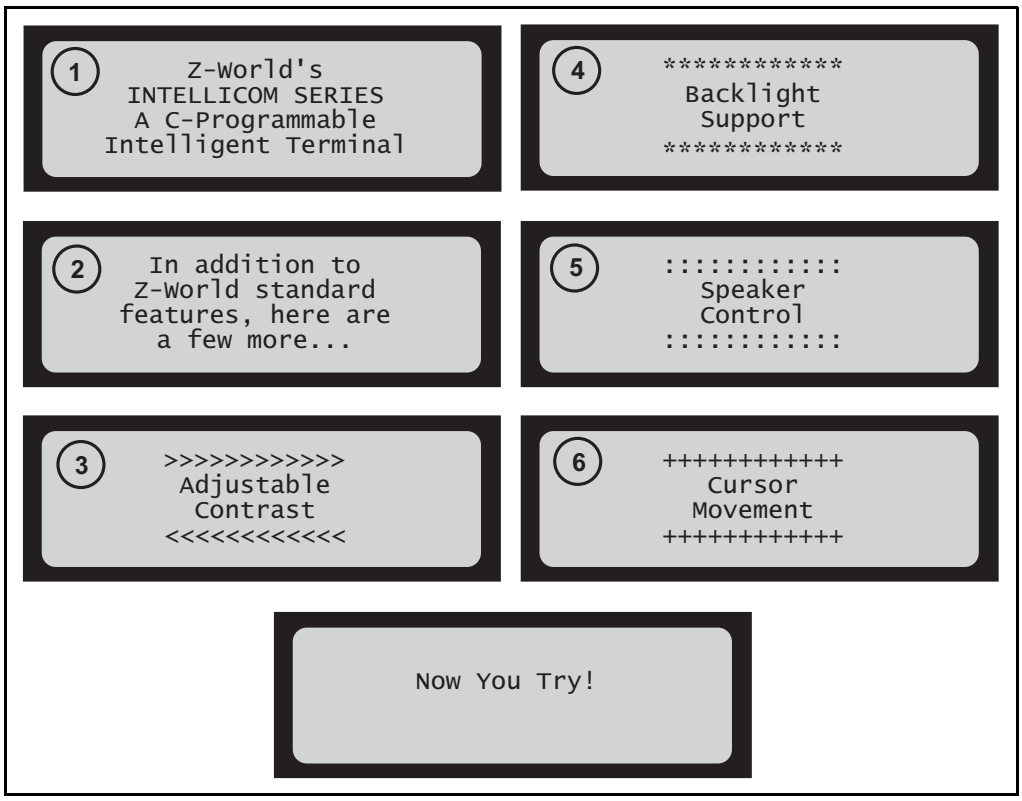
### 4. Apply power.

Plug in the AC adapter. The Intellicom board is now ready to be used.

**NOTE:** A hardware RESET is accomplished by unplugging the AC adapter, then plugging it back in.

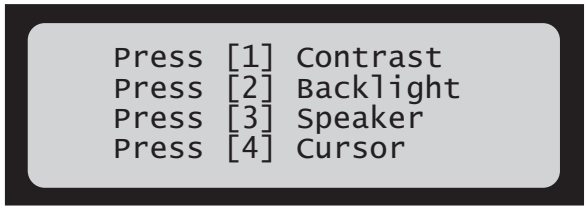
## 2.2 Demonstration Program on Power-Up

The following sequence of messages will be displayed on the LCD when power is first applied to the Intellicom board. Note that the programming cable must ***not*** be connected.

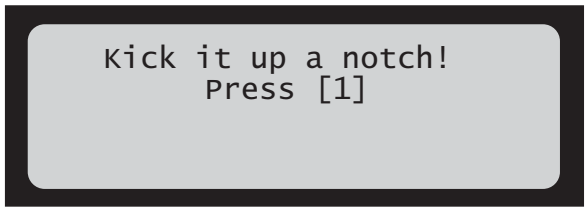


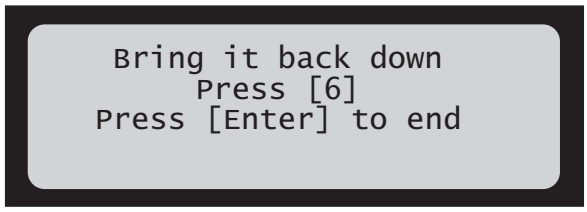
The contrast, backlight, speaker volume, and cursor positions will change automatically through the demonstration. Then there is an opportunity for you to vary these settings by responding to prompts on the LCD.

1. Choose which feature (LCD contrast, backlight on/off, speaker, or cursor) you wish to change.




2. Press [**1**] to select the contrast adjustment demonstration.
3. Press [**1**] to increase contrast, press [**6**] to decrease contrast, or press [**Enter**] to get to choose another feature.

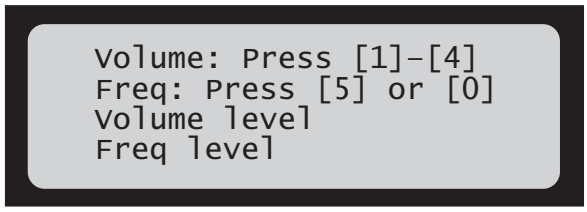


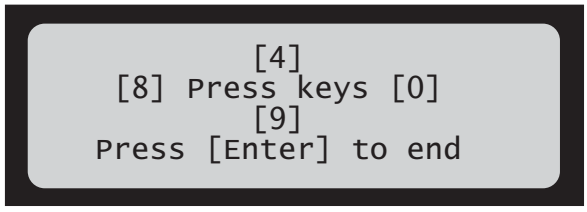

4. Press [**2**] to select the backlight demonstration. Press [**2**] to toggle backlight on or off, or press [**Enter**] to get to choose another feature.




5. Press [**3**] to select the speaker demonstration. Press [**1**]–[**4**] to set the desired speaker volume ([**1**] is min, [**4**] is max), press [**5**] or [**0**] to increase or decrease frequency, or press [**Enter**] to get to choose another feature. The volume and frequency are displayed.




6. Press [**4**] to select the cursor demonstration. Press keys as shown to move cursor, or press [**Enter**] to get to choose another feature.




This demonstration will be replaced by a new program when the programming cable is attached and the new program is compiled and run. The demonstration is available for future reference in the Dynamic C **SAMPLES** directory as **ICOMDEMO.C**.

## 2.3 Programming Cable Connections

### 1. Connect the programming cable to the Intellicom board.

Connect the 10-pin **PROG** connector of the programming cable to header J4 on the Intellicom board as shown in Figure 4. Be sure to orient the red edge of the cable towards pin 1 of the connector. (Do not use the **DIAG** connector, which is used for a normal serial connection.)

**NOTE:** Never disconnect the programming cable by pulling on the ribbon cable. Carefully pull on the connector to remove it from the header.

**NOTE:** Some PCs now come equipped only with a USB port. It may be possible to use an RS-232/USB converter (Part No. 20-151-0178) with the programming cable supplied with the Intellicom Tool Kit. Note that not all RS-232/USB converters work with Dynamic C.

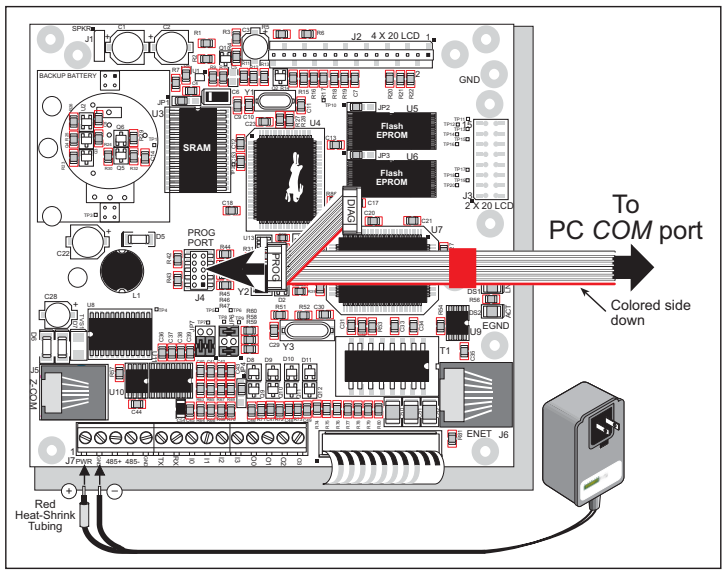



***Figure 4. Programming Cable Connections***

**NOTE:** Be sure to use the programming cable (Part No. 101-0513) supplied with the Intellicom Tool Kit—the programming cable has red shrink wrap around the RS-232 converter section located in the middle of the cable. Programming cables from other Rabbit kits are not designed to work with the Intellicom.

### 2. Apply power.

Reset the Intellicom by unplugging the AC adapter, then plugging it back in. The Intellicom board is now ready to be used.

**NOTE:** A hardware RESET is accomplished by unplugging the AC adapter, then plugging it back in.

To power down the Intellicom, unplug the AC adapter. You should disconnect power before making any circuit adjustments or changing any connections to the board.

## 2.4 Installing Dynamic C

If you have not yet installed Dynamic C, do so now by inserting the Dynamic C CD from the Intellicom Tool Kit in your PC's CD-ROM drive. The CD will auto-install unless you have disabled auto-install on your PC.

If the CD does not auto-install, click **Start > Run** from the Windows **Start** button and browse for the Dynamic C **setup.exe** file on your CD drive. Click **OK** to begin the installation once you have selected the **setup.exe** file.

The online documentation is installed along with Dynamic C, and an icon for the documentation menu is placed on the workstation's desktop. Double-click this icon to reach the menu. If the icon is missing, create a new desktop icon that points to **default.htm** in the **docs** folder, found in the Dynamic C installation folder.

The latest versions of all documents are always available for free, unregistered download from our web sites as well.

The ***Dynamic C User's Manual*** provides detailed instructions for the installation of Dynamic C and any future upgrades.

> **NOTE:** If you have an earlier version of Dynamic C already installed, the default installation of the later version will be in a different folder, and a separate icon will appear on your desktop.

## 2.5 Starting Dynamic C

Once the Intellicom is connected to your PC and to a power source, start Dynamic C by double-clicking on the Dynamic C icon on your desktop or in your **Start** menu.

If you are using a USB port to connect your computer to the Intellicom, choose **Options > Project Options** and select "Use USB to Serial Converter." Click **OK**.

Dynamic C defaults to using the serial port on your PC that you specified during installation. If the port setting is correct, Dynamic C should detect the Intellicom and go through a sequence of steps to cold-boot the Intellicom and to compile the BIOS. (Some versions of Dynamic C will not do the initial BIOS compile and load until the first time you compile a program.)

If you receive the message **`No Rabbit Processor Detected`**, the programming cable may be connected to the wrong COM port, a connection may be faulty, or the target system may not be powered up. First, check both ends of the programming cable to ensure that it is firmly plugged into the PC and the programming port with the marked (colored) edge of the programming cable towards pin 1 of the programming header.

If there are no faults with the hardware, select a different COM port within Dynamic C. From the **Options** menu, 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 active COM port. You should receive a **`Bios compiled successfully`** message once this step is completed successfully.

If Dynamic C appears to compile the BIOS successfully, but you then receive a communication error message when you compile and load a sample program, it is possible that your PC cannot handle the higher program-loading baud rate. Try changing the maximum download rate to a slower baud rate as follows.

- Locate the **Serial Options** dialog in the Dynamic C **Options > Communications** menu. Select a slower Max download baud rate.

If a program compiles and loads, but then loses target communication before you can begin debugging, it is possible that your PC cannot handle the default debugging baud rate. Try lowering the debugging baud rate as follows.

- Locate the **Serial Options** dialog in the Dynamic C **Options > Communications** menu. Choose a lower debug baud rate.

## 2.6 PONG.C

You are now ready to test your set-up by running a sample program.

Find the file **PONG.C**, which is in the Dynamic C **SAMPLES** folder. To run the program, open it with the **File** menu (if it is not still open), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The **STDIO** window will open and will display a small square bouncing around in a box.

This program does not test the serial ports, the I/O, or the TCP/IP part of the board, but does ensure that the board is basically functional. The sample program in Section 5.5, "Run the PINGME.C Demo," tests the TCP/IP portion of the board.

## 2.7 Where Do I Go From Here?

**NOTE:** If you purchased your Intellicom through a distributor or Rabbit partner, contact the distributor or partner first for technical support.

If there are any problems at this point:

- Use the Dynamic C **Help** menu to get further assistance with Dynamic C.
- Check the Rabbit Technical Bulletin Board and forums at www.rabbit.com/support/bb/ and at www.rabbit.com/forums/.
- Use the Technical Support e-mail form at www.rabbit.com/support/.

If the sample program ran fine, you are now ready to go on to explore other Intellicom features and develop your own applications.

Chapter 3, "Subsystems," provides a description of the Intellicom board's features, Chapter 4, "Software," describes the Dynamic C software libraries and introduces some sample programs, and Chapter 5, "Using the TCP/IP Features," explains the TCP/IP features.

# 3. SUBSYSTEMS

Chapter 3 describes the principal subsystems for the Intellicom.

- Intellicom Subsystems
- Serial Communication
- Memory
- Speaker
- Other Hardware

## 3.1 Intellicom Subsystems

Figure 5 shows the Rabbit-based subsystems designed into the Intellicom.

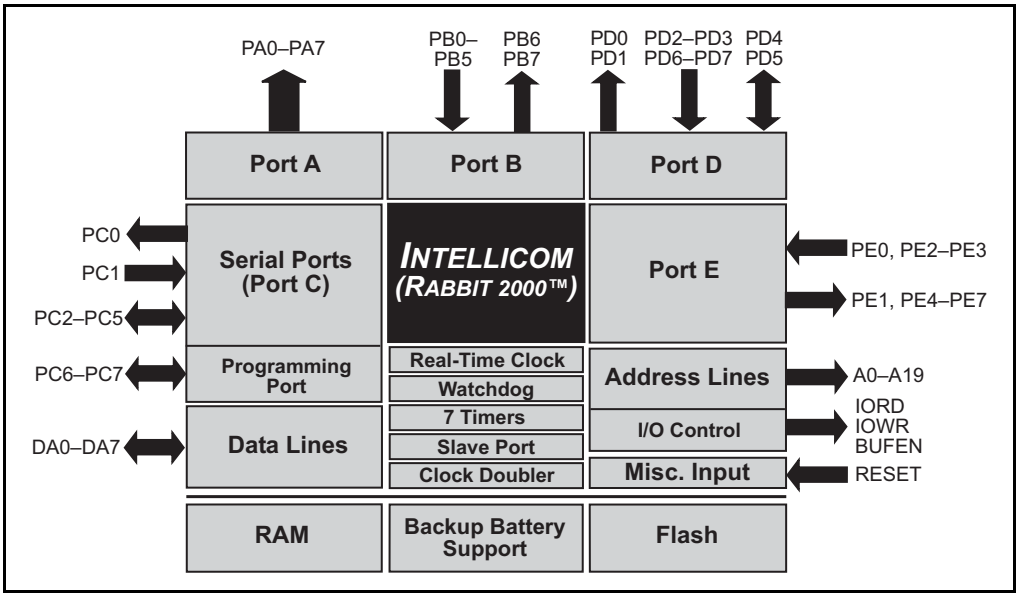



***Figure 5. Intellicom Rabbit-Based Subsystems***

The Intellicom board has 15 pins on header J7, one RJ-12 jack for RS-232 or RS-485 serial communication, and one Ethernet jack (OP6700 only). The pinouts are shown in Figure 6.

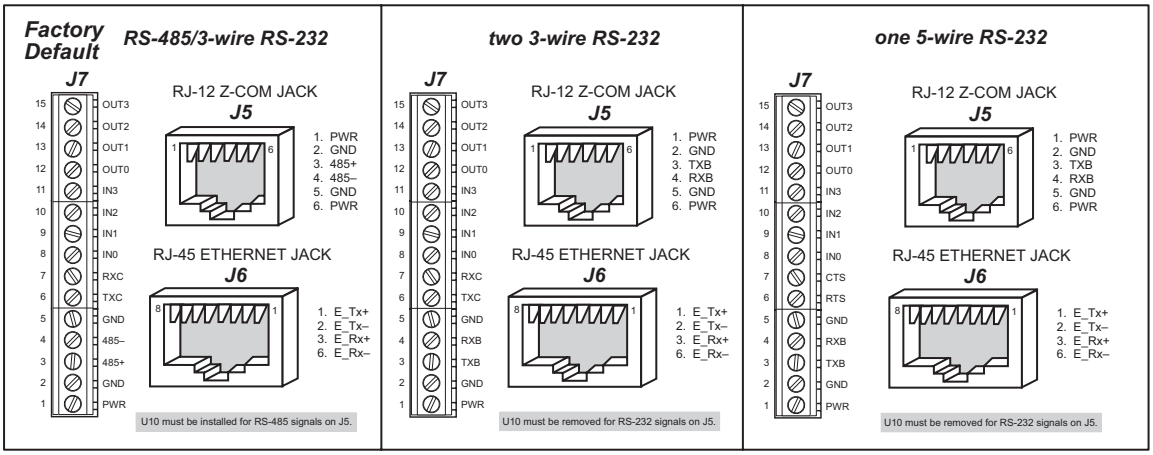



***Figure 6. Intellicom I/O Pinout***

RJ-45 pinouts are sometimes numbered opposite to the way shown in Figure 6. Regardless of the numbering convention followed, the pin positions relative to the spring tab position (located at the bottom of the RJ-45 jack in Figure 6) are always absolute, and the RJ-45 connector will work properly with off-the-shelf Ethernet cables.

### 3.1.1 Digital Inputs

Pins 8–11 on header J7 have the four digital inputs IN0–IN3. Each of the four digital 0 V to 5 V inputs is protected over a range of –36 V to +36 V. The Intellicom is factory-configured for the digital inputs to be pulled up to +5 V, but the digital inputs can also be pulled down by moving the surface-mounted jumper at JP4. The jumper settings and the location of JP4 are shown in Figure 7.

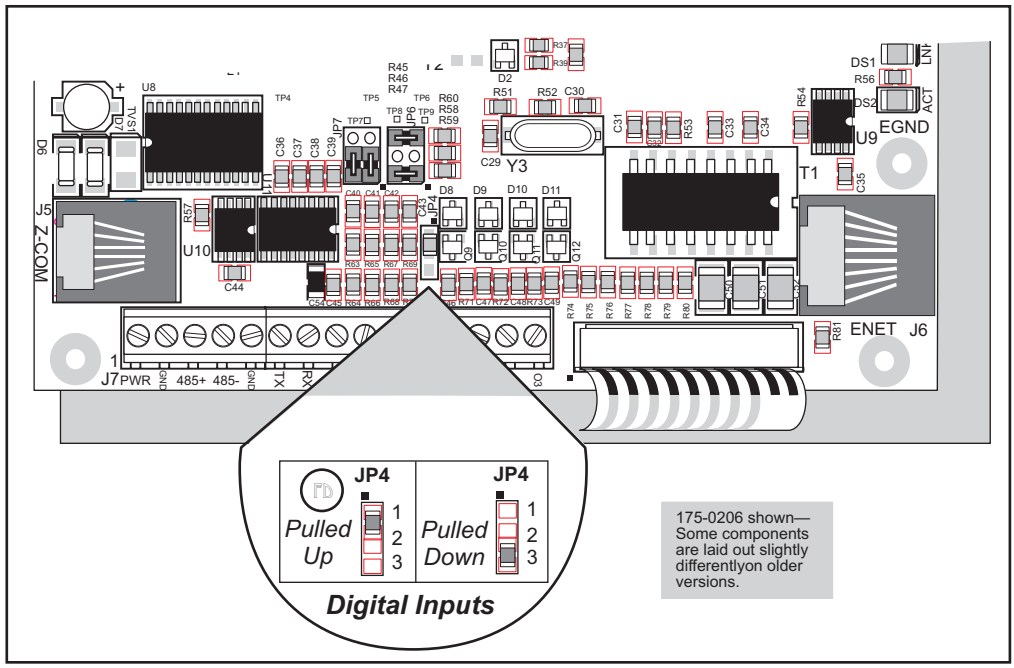



***Figure 7. Surface-Mounted Jumper Configurations for Selecting Pullup/Pulldown on the Digital Inputs***

### 3.1.2 Digital Outputs

Pins 12–15 on header J7 have the four digital outputs OUT0–OUT3. Each of the four open-collector digital outputs can sink up to 200 mA at 40 V DC.

## 3.2 Serial Communication

In the factory-default configuration, the Intellicom has one RS-232 (3-wire) serial channel, one RS-485 serial channel, and one synchronous CMOS serial channel. The Intellicom may be configured for 5-wire RS-232 or two 3-wire RS-232 channels. The exact configuration instructions depend on the version of Intellicom board you have. This information is etched on the bottom side of the printed circuit board, or you can readily determine your version by examining the diagrams below to find the one that matches your board.

***Version 175-0188 Rev. A & B***

The RS-232 transceiver may be used as a 5-wire RS-232 channel or as two 3-wire RS-232 channels at the expense of the RS-485 channel by adding 0 Ω surface-mounted resistors at R61 and R62 as shown in Figure 8(a). The RS-485 chip (U10) and the associated bias and termination resistors (R58, R59, and R60) shown in Figure 9(a) must be removed when configuring the Intellicom for either one 5-wire RS-232 or two 3-wire RS-232.

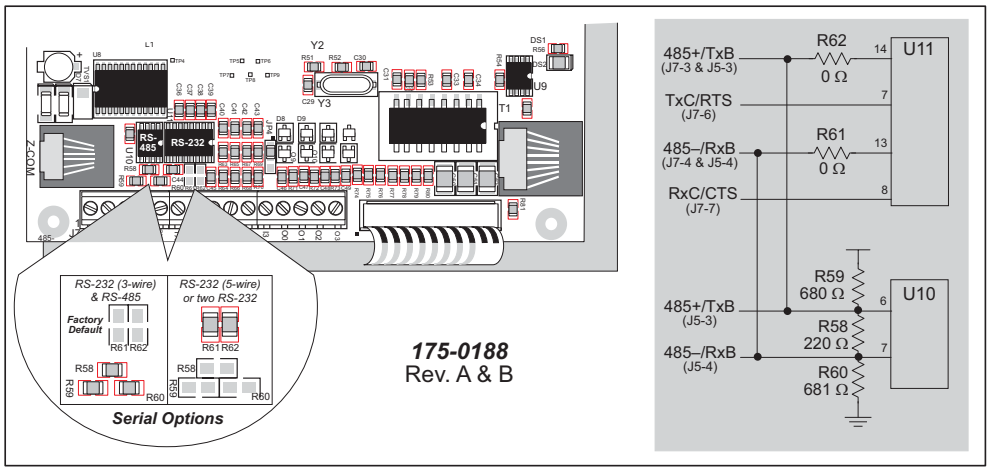



***Figure 8(a). Intellicom RS-232/RS-485 Serial Communication Options***

Table 2(a) summarizes the options. Note that the parameters in the **`serMode`** software function call must also be set to match the hardware configuration being used.

***Table 2(a). Serial Communication Configurations (Version 175-0188 Rev. A & B)***

| Item | One 3-wire RS-232 & RS-485 | Two 3-wire RS-232 | One 5-wire RS-232 |
|---|---|---|---|
| R58–R60 | In | Out | Out |
| R61–R62 | Out | In | In |
| U10 | In | Out | Out |
| J7-3 & J5-3 | RS-485+ | TxB | TxB |
| J7-4 & J5-4 | RS-485– | RxB | RxB |
| J7-6 | TxC | TxC | RTS |
| J7-7 | RxC | RxC | CTS |

***Version 175-0188 Rev. C***

The RS-232 transceiver may be used as a 5-wire RS-232 channel or as two 3-wire RS-232 channels at the expense of the RS-485 channel, which is connected through 0 Ω surface-mounted resistors at R82 and R83 as shown in Figure 8(b). R82 and R83, shown in Figure 8(b), must be removed when configuring the Intellicom for either one 5-wire RS-232 or two 3-wire RS-232. U10 and the associated bias and termination resistors (R58, R59, and R60) must also be removed, but R82 and R83 are left installed, if you wish the TxB and RxB RS-232 signals to be available on header J5.

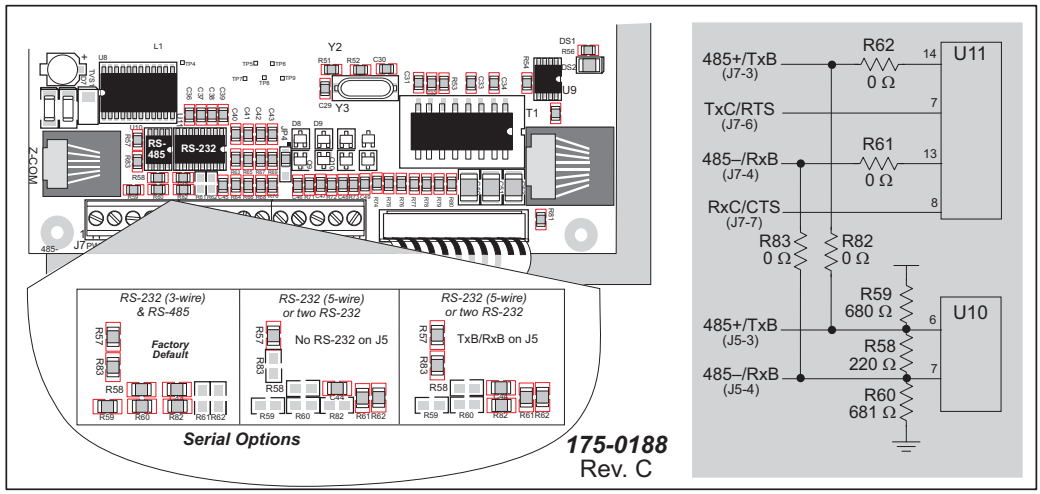



***Figure 8(b). Intellicom RS-232/RS-485 Serial Communication Options***

Table 2(b) summarizes the options. Note that the parameters in the **`serMode`** software function call must also be set to match the hardware configuration being used.

***Table 2(b). Serial Communication Configurations (Version 175-0188 Rev. C)***

| Item | One 3-wire RS-232 & RS-485 | Two 3-wire RS-232 | One 5-wire RS-232 | RS-232 on J5 |
|---|---|---|---|---|
| R58–R60 | In | — | — | Out |
| R61–R62 | Out | In | In | In |
| R82–R83 | In | Out | Out | In |
| U10 | In | In | In | Out |
| J7-3 | RS-485+ | TxB | TxB | TxB |
| J7-4 | RS-485– | RxB | RxB | RxB |
| J7-6 | TxC | TxC | RTS | TxC or RTS |
| J7-7 | RxC | RxC | CTS | RxC or CTS |
| J5-3 | RS-485+ | — | — | TxB |
| J5-4 | RS-485– | — | — | RxB |

### *Version 175-0206*

The RS-232 transceiver may be used as a 5-wire RS-232 channel or as two 3-wire RS-232 channels at the expense of the RS-485 channel, which is connected through jumpers across header JP7 as shown in Figure 8(c). The jumper configurations are shown in Figure 8(c).

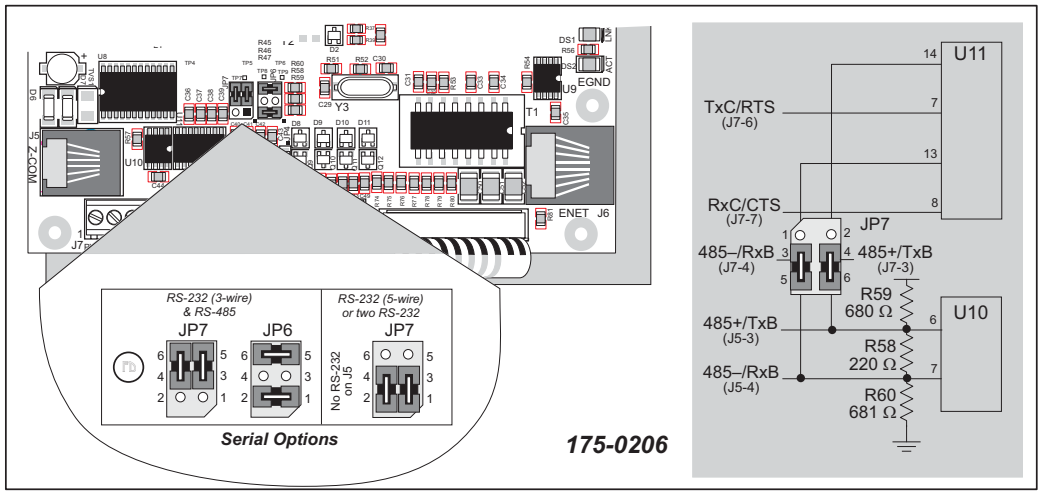



***Figure 8(c). Intellicom RS-232/RS-485 Serial Communication Options***

Table 2(c) summarizes the options. Note that the parameters in the **`serMode`** software function call must also be set to match the hardware configuration being used.

***Table 2(c). Serial Communication Configurations (Version 175-0206)***

| Item | One 3-wire RS-232 & RS-485 | Two 3-wire RS-232 | One 5-wire RS-232 | RS-232 on J5 |
|---|---|---|---|---|
| Header JP7 | 3–5<br>4–6 | 1–3<br>2–4 | 1–3<br>2–4 | 1–5<br>2–6 |
| Header JP6 | 1–2<br>5–6 | — | — | No jumpers installed |
| U10 | In | In | In | Out |
| J7-3 | RS-485+ | TxB | TxB | — |
| J7-4 | RS-485– | RxB | RxB | — |
| J7-6 | TxC | TxC | RTS | TxC or RTS |
| J7-7 | RxC | RxC | CTS | RxC or CTS |
| J5-3 | RS-485+ | — | — | TxB |
| J5-4 | RS-485– | — | — | RxB |

### 3.2.1 RS-232

The Intellicom's RS-232 serial channel is connected to an RS-232 transceiver, U11. U11 provides the voltage output, slew rate, and input voltage immunity required to meet the RS-232 serial communication protocol. Basically, the chip translates the Rabbit 2000's 0 V to +Vcc signals to RS-232 signal levels. Note that the polarity is reversed in an RS-232 circuit so that +5 V is output as approximately -10 V and 0 V is output as approximately +10 V. U11 also provides the proper line loading for reliable communication.

The maximum baud rate is 115,200 bps. RS-232 can be used effectively at this baud rate for distances up to 15 m.

### 3.2.2 RS-485

The Intellicom has one RS-485 serial channel, which is connected to the Rabbit 2000 serial port B through U10, an RS-485 transceiver. The chip's slew rate limiters provide for a maximum baud rate of 250,000 bps, which allows for a network of up to 1200 m (or 4000 ft). The half-duplex communication uses the Rabbit 2000's PC0 pin to control the data enable on the communication line.

The RS-485 signals are available on pins 3 and 4 of header J7, and on J5, the RJ-12 jack.

The Intellicom can be used in an RS-485 multidrop network. Connect the 485+ to 485+ and 485– to 485– using single twisted-pair wires (nonstranded, tinned) as shown in Figure 9.

Alternatively, the RS-485 multidrop network may be hooked up using cables with RJ-12 plugs. Note that the RJ-12 jack has +RAW_485 and GND, which means that only ***one*** Intellicom needs to be connected to an external power source via an AC adapter. When doing so, ensure that the AC adapter has sufficient capacity for the network — each Intellicom unit nominally draws 100 mA at 24 VDC.



**CAUTION:** If you plan to connect a power supply to more than one Intellicom unit in an RS-485 network using the RJ-12 jacks, rework the RS-485 cables so they do not connect +RAW_RS485 through the RJ-12 jack to the boards in the network.

**NOTE:** The RS-485 port is available only in the factory default configuration. The RS-485 port will not be available when you select the configuration option for both 3-wire RS-232 ports or one 5-wire RS-232 port.

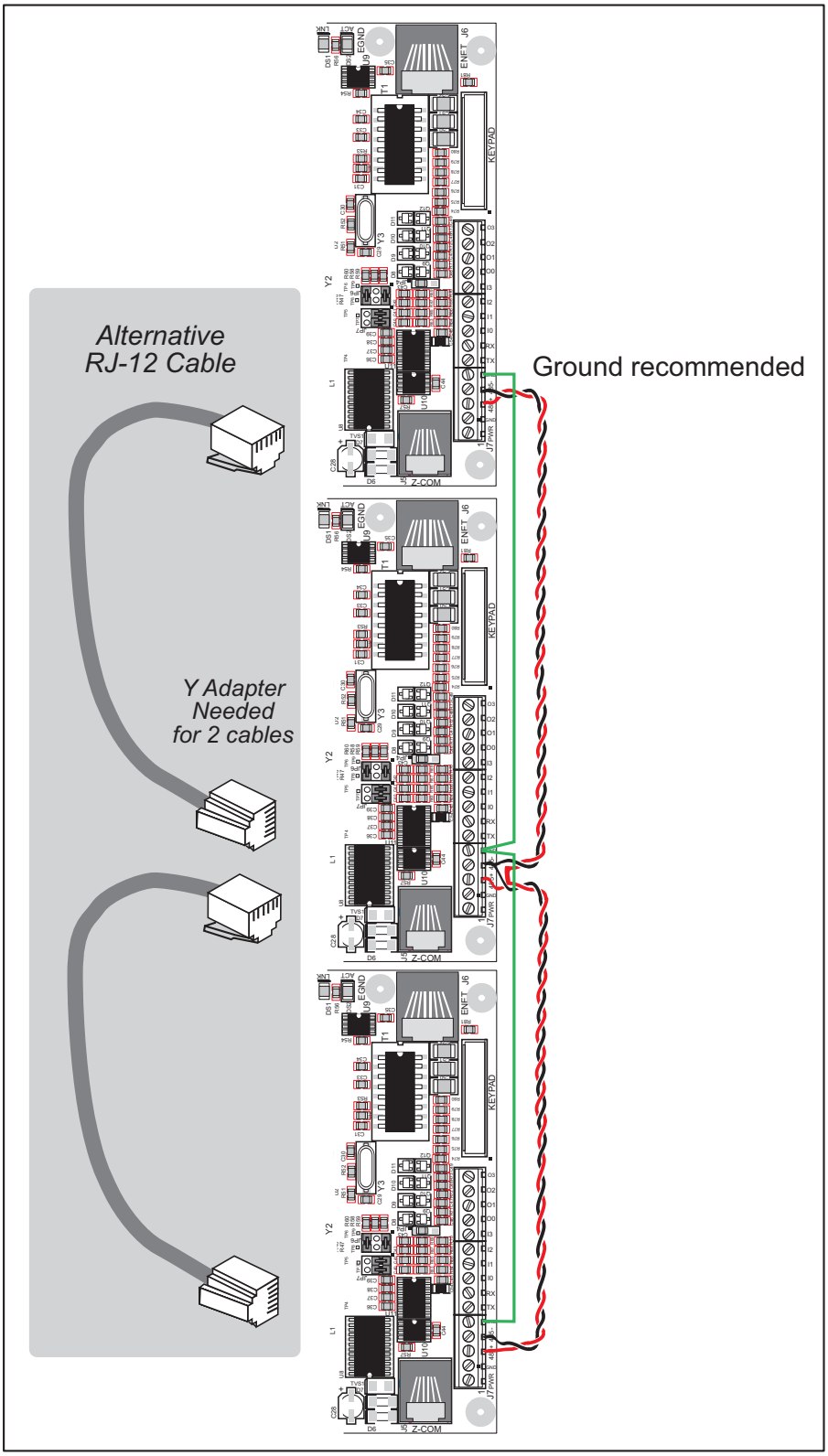



***Figure 9. Multidrop Intellicom Network***

The Intellicom comes with a 220 Ω termination resistor and two 680 Ω bias resistors installed and enabled with jumpers across pins 1–2 and 5–6 on header JP6, as shown in Figure 10.

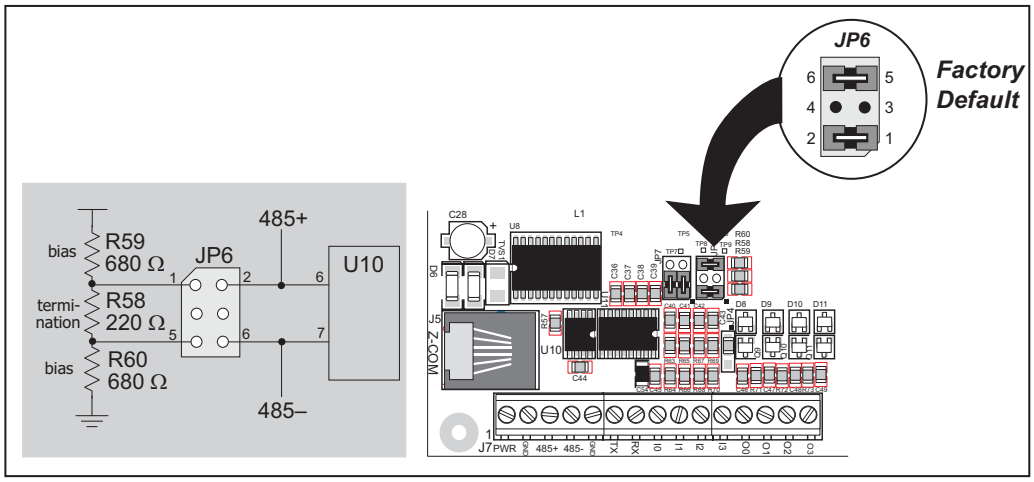



***Figure 10. RS-485 Termination and Bias Resistors***

The bias and termination resistors in a multidrop network should only be enabled on both end nodes of the network. Disable the termination and bias resistors on the intervening Intellicom units in the network by removing both jumpers from header JP6. Note that older versions of the Intellicom do not have this jumper feature, and the surface-mounted bias and termination resistors shown in Figure 10 have to be removed in networks containing more than 10 Intellicom units.

### 3.2.3 Programming Port

The Intellicom has a 10-pin programming header labeled J4. The programming port uses the Rabbit 2000's serial port A for communication. The programming port uses the Rabbit 2000's Serial Port A for communication. 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 2000 on the Intellicom 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.

#### 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 input

The programming port may also be used as a serial port via the **DIAG** connector on the programming cable.

In addition to Serial Port A, the Rabbit 2000 startup-mode (SMODE0, SMODE1), status, and reset pins are available on the programming port header.

The two startup mode pins determine what happens after a reset—the Rabbit 2000 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 output.

The /RESET_IN pin is an external input that is used to reset the Rabbit 2000 and the Intellicom onboard peripheral circuits.

Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information.

## 3.3 Programming Cable

The programming cable is used to connect the Intellicom tprogramming port o a PC serial COM port. The programming cable converts the RS-232 voltage levels used by the PC serial port to the voltage levels used by the Rabbit 2000.

When the **PROG** connector on the programming cable is connected to the Intellicom programming port, programs can be downloaded and debugged over the serial interface between the PC and the Rabbit 2000.

The **DIAG** connector of the programming cable may be used on header J4 of the Intellicom with the Intellicom operating in the Run Mode. This allows the programming port to be used as a regular serial port.

### 3.3.1 Changing Between Program Mode and Run Mode

The Intellicom is automatically in Program Mode when the **PROG** connector on the programming cable is attached, and is automatically in Run Mode when no programming cable is attached. When the Rabbit 2000 is reset, the operating mode is determined by the status of the SMODE pins. When the programming cable's **PROG** connector is attached, the SMODE pins are pulled high, placing the Rabbit 2000 in the Program Mode. When the programming cable's **PROG** connector is not attached, the SMODE pins are pulled low, causing the Rabbit 2000 to operate in the Run Mode.

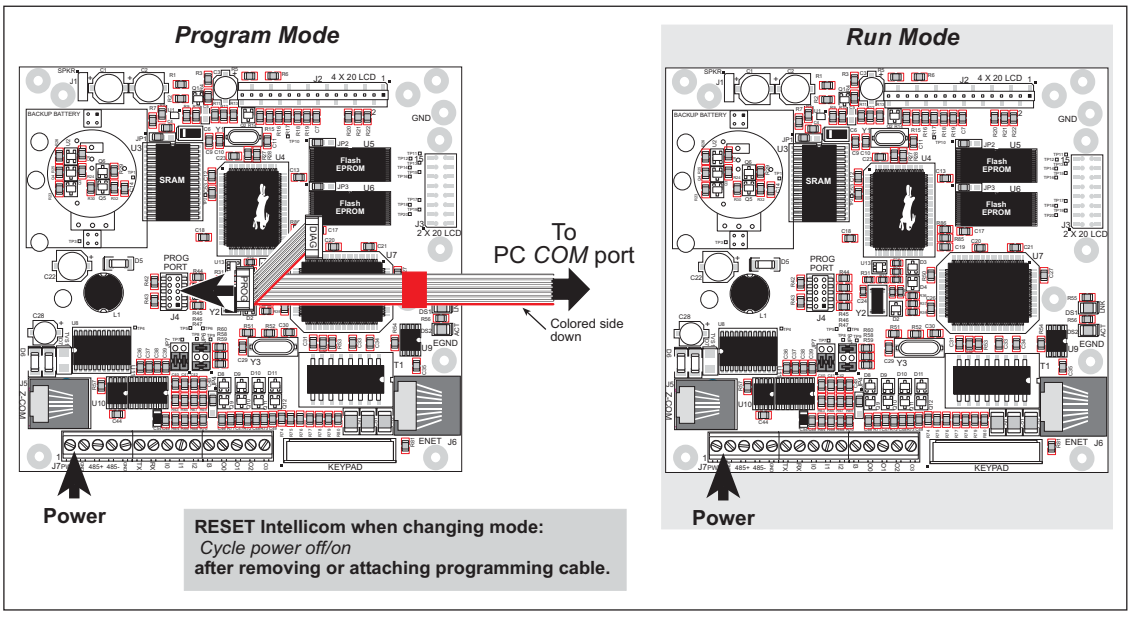



***Figure 11. Intellicom Program Mode and Run Mode Set-Up***

A program "runs" in either mode, but can only be downloaded and debugged when the OP6800 is in the Program Mode.

Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information on the programming port and the programming cable.

## 3.4 Memory

### 3.4.1 SRAM

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

The standard models come with 128K of SRAM. Figure 12 shows the locations and the jumper settings for the jumpers at JP1 used to set the SRAM size. The "jumpers" are 0 Ω surface-mounted resistors.

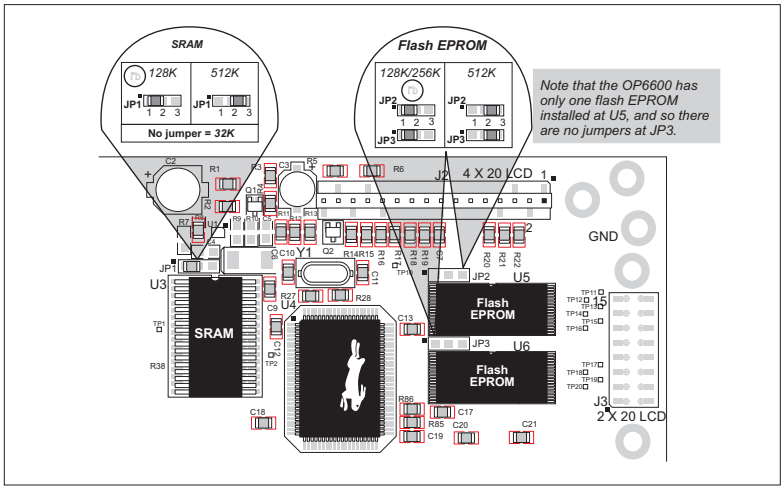



***Figure 12. Intellicom Jumper Settings for SRAM and Flash EPROM Size***

### 3.4.2 Flash Memory

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

The Intellicom OP6700 comes with two 256K flash memory chips, and the Intellicom OP6600 comes with one 256K flash memory. Figure 12 shows the locations and the jumper settings for the jumpers at JP2 and JP3 used to set the flash memory size. The "jumpers" are 0 Ω surface-mounted resistors.

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

A Flash Memory Bank Select jumper configuration option exists at JP5 with 0 Ω surface-mounted resistors. This option, used in conjunction with some configuration macros, allows Dynamic C to compile two different co-resident programs for the upper and lower halves of the 256K flash in such a way that both programs start at logical address 0000. This is useful for applications that require a resident download manager and a separate downloaded program. See Technical Note 218, ***Implementing a Serial Download Manager for a 256K Flash***, for details.

### 3.4.3 Dynamic C BIOS Source Files

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

## 3.5 Speaker

The Intellicom comes with a 35 Ω speaker that is controlled through the Dynamic C function **spkrOut**. Both the volume and the frequency of the signal are set with this function call. The maximum average volume was measured to be 75 dBA @ 30 cm (12 inches) from the speaker. Figure 13 shows typical volume measurements for various frequencies with the speaker grille open and closed to maintain water resistance for the front mounting panel.

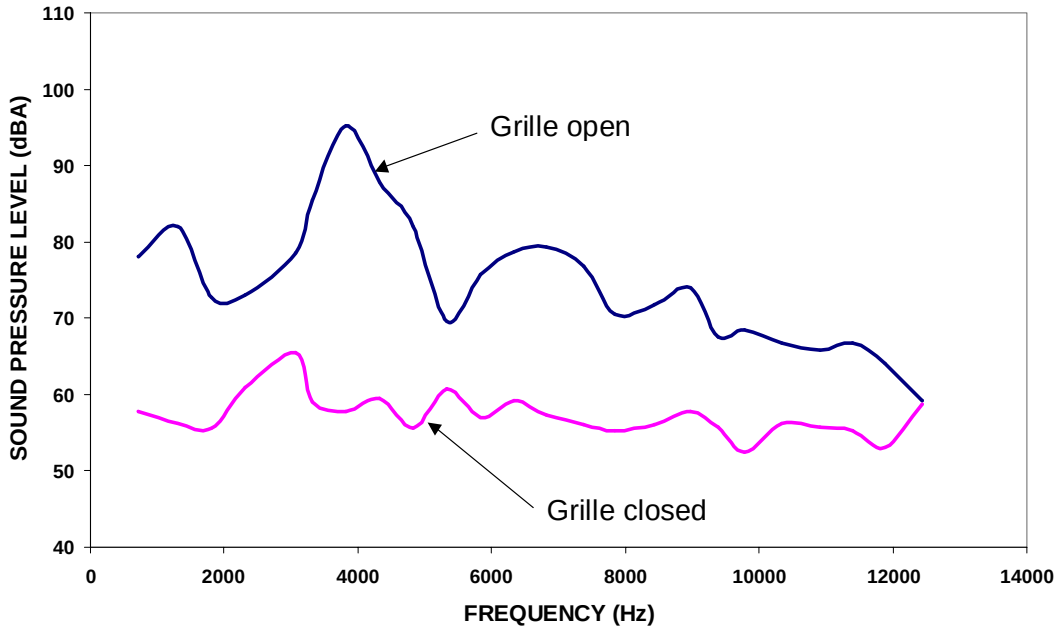



***Figure 13. Intellicom Speaker Sound Pressure Level 30 cm (12 inches) from Speaker***

## 3.6 Vacuum Fluorescent Display

A vacuum fluorescent display (VFD) may be substituted for the LCD by removing R22 and substituting a VFD for LCD. Note that a VFD has no backlighting and no contrast control.

**NOTE:** Contact your Rabbit sales representative for information on ordering this option from the factory.

The instructions for accessing the display are similar to those for accessing the keypad insert in Appendix B, "Keypad and Plastic Enclosure."

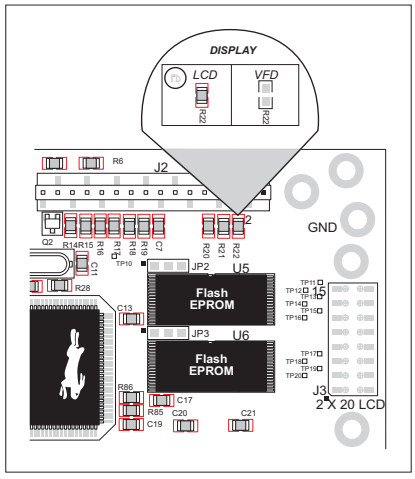



***Figure 14. Location of Display Control Resistor, R22***

## 3.7 Other Hardware

### 3.7.1 Clock Doubler

The Intellicom uses an 18.432 MHz crystal, and so its clock doubler is disabled automatically in the BIOS.

### 3.7.2 Spectrum Spreader

Intellicom boards that carry the CE mark have a Rabbit 2000 microprocessor that features a spectrum spreader, which helps to mitigate EMI problems. By default, the spectrum spreader is on automatically for Intellicom boards that carry the CE mark when used with Dynamic C 7.32 or later versions, but the spectrum spreader may also be turned off or set to a stronger setting. The means for doing so is through a simple global 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 needed for the Intellicom.
3. Click **OK** to save the macro. The spectrum spreader will now remain off whenever you are in the project file where you defined the macro.

There is no spectrum spreader functionality for Intellicom boards that do not carry the CE mark or when using any Intellicom with a version of Dynamic C prior to 7.30.

# 4. SOFTWARE

Dynamic C is an integrated development system for writing embedded software. It runs on an IBM-compatible PC and is designed for use with controllers based on the Rabbit® microprocessor.

Chapter 4 provides the libraries, function calls, and sample programs related to the Intellicom.

## 4.1 Running Dynamic C

You have a choice of doing your software development in the flash memory or in the static RAM included on the Intellicom. The flash memory and SRAM options are selected with the **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 can be developed in RAM, but cannot run standalone from RAM after the programming cable is disconnected. All standalone applications can only run from flash memory.

**NOTE:** Do not depend on the flash memory sector size or type. Due to the volatility of the flash memory market, the Intellicom and Dynamic C were designed to accommodate flash devices with various sector sizes.

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 a binary image file for later loading. Dynamic C runs on PCs under Windows 95 or later. Programs can be downloaded at baud rates of up to 230,000 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.

### 4.1.1 Upgrading Dynamic C

#### 4.1.1.1 Patches and Bug Fixes

Dynamic C patches that focus on bug fixes are available from time to time. Check the Web site www.rabbit.com/support/ for the latest patches, workarounds, and bug fixes.

The default installation of a patch or bug fix is to install the file in a directory (folder) different from that of the original Dynamic C installation. Rabbit recommends using a different directory so that you can verify the operation of the patch without overwriting the existing Dynamic C installation. If you have made any changes to the BIOS or to libraries, or if you have programs in the old directory (folder), make these same changes to the BIOS or libraries in the new directory containing the patch. Do ***not*** simply copy over an entire file since you may overwrite a bug fix. Once you are sure the new patch works entirely to your satisfaction, you may retire the existing installation, but keep it available to handle legacy applications.

#### 4.1.1.2 Upgrades

Dynamic C software supplied with the Tool Kit is designed for use with the board it is included with, and is included at no extra charge. Dynamic C is a complete software development system, but does not include all of Dynamic C's features and upgrade path. These extra features include the popular µC/OS-II real-time operating system, as well as PPP, Advanced Encryption Standard (AES), and other select libraries. A one-year maintenance agreement for telephone tech support and an upgrade path for all new Dynamic C releases is also available.

## 4.2 Sample Programs

Sample programs are provided in the Dynamic C **SAMPLES** folder. The sample program **PONG.C** demonstrates the output to the **STDIO** window. The various directories in the **SAMPLES** folder contain specific sample programs that illustrate the use of the corresponding Dynamic C libraries.

The **SAMPLES\ICOM** and **SAMPLES\TCPIP** folders provides sample programs specific to the Intellicom board. 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), compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. The Intellicom must be in **Program** mode (see Section 3.3, "Programming Cable,") and must be connected to a PC using the programming cable as described in Section 2.3, "Programming Cable Connections."

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

### 4.2.1 General Intellicom Operation

- **COFTERMA.C**—Demonstrates cofunctions, the cofunction serial library, and using a serial ANSI terminal such as Hyperterminal from an available PC COM port connection.
- **DEMOBRD1.C**—Flashes LEDs on Demonstration Board included in Development Kit. See Appendix D for hookup instructions for the Demonstration Board.
- **DEMOBRD2.C**—Flashes LEDs on Demonstration Board included in Development Kit and illustrates the Dynamic C **runwatch** function. See Appendix D for hookup instructions for the Demonstration Board.
- **DEMOBRD3.C**—Flashes LEDs on Demonstration Board included in Development Kit and demonstrates the use of costatements. See Appendix D for hookup instructions for the Demonstration Board.
- **HTTPDEMO.C**—Allows a Web browser to view and change the state of the Intellicom board. See Appendix D for hookup instructions for the Demonstration Board.
- **ICOM232.C**—Demonstrates a simple RS-232 loopback.
- **ICOM485.C**—Demonstrates a simple RS-485 transmission from master to slave.
- **ICOM5WIRE.C**—Demonstrates a 5-wire RS-232 loopback in an Intellicom set up for 5-wire RS-232.
- **ICOMDEMO.C**—Demonstration program to illustrate Intellicom features. This demonstration program comes up when the Intellicom is first powered up before new programs are compiled and run.
- **ICOMIO.C**—Demonstrates how to turn the digital I/O on and off.
- **KEYLCD.C**—Demonstrates use of LCD and keypad.

- **MBOXDEMO.C**—Implements a Web server that allows e-mail messages to be entered and then shown on the LCD display. See Appendix D for hookup instructions for the Demonstration Board.
- **MUSIC.C**—Speaker demonstration: plays one line of "Bicycle Built For Two" (with lyrics).
- **MUSIC2.C**—Speaker demonstration: plays one line of "Für Elise" as background music while other processing is going on.
- **REMOTE1.C**—Demonstrates simple serial data communication using a remote ANSI terminal such as Hyperterminal from an available PC COM port connection.
- **SMTPDEMO.C**—Uses the **TCPIP\SMTP.LIB** library to send an e-mail when a key on the keypad or a switch on the Demonstration Board is pressed. See Appendix D for hookup instructions for the Demonstration Board.
- **SPEAKER.C**—Demonstrates how to adjust the speaker frequency and volume.
- **TCP_RESPOND.C**—Shows how to receive messages and respond.
- **TCP_SEND.C**—Shows how to send message to specific addresses and ports.

The programs **TCP_SEND.C** and **TCP_RESPOND.C** are meant to be executed on two different Intellicom boards so that the two boards communicate with each other. In the absence of a second board, **PCSEND.EXE** (used with **TCP_SEND.C**) and **PCRESPOND.EXE** (used with **TCP_RESPOND.C**) in the **SAMPLES\ICOM\WINDOWS** directory can be used on the PC console side at the command prompt. Both the executables and the C source code are located in the **WINDOWS** directory.

### Using PCSEND

**PCSEND.C** is the source code for **PCSEND.EXE** used on the PC console side to communicate with an Intellicom board. The executable **PCSEND.EXE** is similar to **TCP_SEND.C**, but is run at the command prompt to communicate with an Intellicom board running **TCP_RESPOND.C**.

### Using PCRESPOND

**PCRESPOND.C** is the source code for **PCRESPOND.EXE** used on the PC console side to communicate with an Intellicom board. The executable **PCRESPOND.EXE** is similar to **TCP_RESPOND.C**, but is run at the command prompt to communicate with an Intellicom board running **TCP_SEND.C**.

## 4.3 Dynamic C Libraries

Two library directories are specific to the Intellicom.

- **ICOM**—libraries associated with features specific to the Intellicom unit.
- **TCPIP**—libraries specific to using TCP/IP functions on the Intellicom board.

Other functions applicable to all devices based on the Rabbit microprocessors are described in the ***Dynamic C Function Reference Manual***.

The TCP/IP functions are described in the ***Dynamic C TCP/IP User's Manual*** included in the manual set with the ***Dynamic C User's Manual***. Additional TCP/IP libraries are added on an ongoing basis.

## 4.4 Intellicom Function Calls

### 4.4.1 Board Initialization

```
void brdInit (void);
```

Initializes port registers for the operation of the board. Call this function at the beginning of the application.

**RETURN VALUE**

None

**SEE ALSO**

`dispInit, keyInit`

### 4.4.2 Digital I/O

```
void digOut (int channel, int value);
```

Sets the state of a digital output.

**PARAMETERS**

**`channel`** is the output channel number (0, 1, 2, or 3).

**`value`** is the output value (0 or 1).

**RETURN VALUE**

None.

**SEE ALSO**

`digIn`

```
int digIn (int channel);
```

Reads the state of a digital input.

**PARAMETERS**

**`channel`** is the input channel number (0, 1, 2, or 3).

**RETURN VALUE**

The state of the input (0 or 1).

**SEE ALSO**

`digOut`

### 4.4.3 Serial Communication

Library files included with Dynamic C provide a full range of serial communications support. The **RS232.LIB** library provides a set of circular-buffer-based serial functions. The **PACKET.LIB** library provides packet-based serial functions where packets can be delimited by the 9th bit, by transmission gaps, or with user-defined special characters. Both libraries provide blocking functions, which do not return until they are finished transmitting or receiving, and nonblocking functions, which must be called repeatedly until they are finished. For more information, see the ***Dynamic C Function Reference Manual*** and Technical Note 213, ***Rabbit 2000 Serial Port Software***.

The following function calls are specific to the Intellicom.

```
int serMode (int mode);
```

User interface to set up serial communication lines for the Intellicom board. Call this function after **serXOpen()**.

**PARAMETERS**

**mode** is the defined serial port configuration of the devices installed.

| Mode | Serial Port | |
|---|---|---|
| | B | C |
| 0 | RS-485 | RS-232, 3-wire |
| 1 | RS-232, 3-wire | RS-232, 3-wire |
| 2 | RS-232, 5-wire | CTS/RTS |

**RETURN VALUE**

0 if correct mode, 1 if not.

**SEE ALSO**

**serB485Tx, serB485Rx**

```
void serB485Tx();
```

Sets pin 3 (DE) high to disable Rx and enable Tx.

**SEE ALSO**

**serMode, serB485Rx**

```
void serB485Rx();
```

Resets pin 3 (DE) low to enable Rx and disable Tx.

**SEE ALSO**

**serMode, serB485Tx**

### 4.4.4 Keypad Controls

```
void keyProcess (void);
```

Scans and processes keypad data (up to 8 × 8 matrix) for key assignment, debouncing, press and release, and repeat. Provides debouncing, user-definable key code, separate press and release code (both optional), two- and three-speed auto repeat.

**RETURN VALUE**

None.

**SEE ALSO**

`keyConfig, keyGet, keypadDef`

```
void keyConfig (char cRaw, char cPress,
   char cRelease, char cCntHold, char cSpdLo,
   char cCntLo, char cSpdHi);
```

Assigns user-defined keys to keypad positions. Defines ticks for key debouncing and speed.

**PARAMETERS**

**`cRaw`** is the Raw Key Code Index, a 2 × 6 keypad matrix with the following raw keycode index assignments.

| | Col 5 | Col 4 | Col 3 | Col 2 | Col 1 | Col 0 |
|---|---|---|---|---|---|---|
| **Row 1** | 5 | 4 | 3 | 2 | 1 | 0 |
| **Row 0** | 13 | 12 | 11 | 10 | 9 | 8 |

**`cPress`** is the Key Press Code, an 8-bit value returned when a key is pressed (0 = Unused). See **`keypadDef`** for default press codes.

**`cRelease`** is the Key Release Code, an 8-bit value to be returned when a key is released (0 = Unused).

**`cCntHold`** is Hold Ticks, how long to hold before repeating (0 = No Repeat).

**`cSpdLo`** is Low-Speed Repeat Ticks, how many times to repeat (0 = None).

**`cCntLo`** is Low-Speed Hold Ticks, how long to hold before going to high-speed repeat (0 = Slow Only).

**`cSpdHi`** is High-Speed Repeat Ticks, how many times to repeat after low-speed repeat (0 = None).

**RETURN VALUE**

None.

**SEE ALSO**

`keyProcess, keyGet, keypadDef`

## char keyGet (void);

Gets next keypress.

**RETURN VALUE**

The next keypress, or 0 if none.

**SEE ALSO**

**keyConfig, keyProcess, keypadDef**

## void keyInit (void);

Initializes keypad process.

**RETURN VALUE**

None.

**SEE ALSO**

**brdInit**

## void keypadDef();

Configures keypad to default layout:

[ 1 ][ 2 ][ 3 ][ 4 ][ 5 ][ . ]
[ 6 ][ 7 ][ 8 ][ 9 ][ 0 ][Enter]

**RETURN VALUE**

None.

**SEE ALSO**

**keyConfig, keyGet, keyProcess**

### 4.4.5 Display Controls

## void dispContrast (char vcontrast);

Sets display contrast.

**PARAMETERS**

Suggested values are 40–128 for high to low contrast, although 4–252 can be used. Contrast adjustment not supported on VFDs.

**RETURN VALUE**

None.

**SEE ALSO**

**dispOnoff, dispBacklight**

## void dispCursor (unsigned int wStyle);

Sets cursor type: on, off, or blinking.

**PARAMETERS**

**wStyle** is one of the following cursor macros:

```
DISP_CUROFF      for cursor off
DISP_CURON       for cursor on
DISP_CURBLINK     for cursor blink
```

**RETURN VALUE**

None.

**SEE ALSO**

**dispClear, dispGoto**

## void dispGoto (unsigned wX, unsigned wY);

Positions the cursor.

**PARAMETERS**

**wX** is the column position, 0 to 19.

**wY** is the row position, 0 to 3.

**RETURN VALUE**

None.

**SEE ALSO**

**dispClear, dispCursor**

## void dispClear (void);

Clears the display and homes cursor.

**RETURN VALUE**

None.

**SEE ALSO**

`dispGoto, dispCursor`

## void dispPutc (char cByte);

Puts a character on the display.

**PARAMETER**

**`cByte`** is the character to display.

**RETURN VALUE**

None.

**SEE ALSO**

`dispPrintf`

## void dispPrintf (char *pcFormat, ...);

Prints formatted string to the display, similar to **`printf`** call.

**PARAMETER**

**`pcFormat`** is the formatted string.

**RETURN VALUE**

None.

**SEE ALSO**

`dispPutc`

## void dispOnoff (int onOff);

Turns the display on or off.

**PARAMETERS**

Set or write 1 to turn the display on. Clear or write 0 to turn the display off.

**RETURN VALUE**

None.

**SEE ALSO**

`dispContrast, dispBacklight`

## void dispBacklight (int onOff);

Sets the backlight on or off. Not supported on VFDs.

**PARAMETERS**

Set or write 1 to turn the backlight on. Clear or write 0 to turn the backlight off.

**RETURN VALUE**

None.

**SEE ALSO**

`dispContrast, dispOnoff`

## void dispInit();

Initializes the display. Specifically, the function reinitializes the display controller, clears the display, and puts a nonblinking underline cursor in the top left position.

**RETURN VALUE**

None.

**SEE ALSO**

`brdInit`

### 4.4.6 Speaker Controls

```
void spkrOut (unsigned int wFreq, unsigned int wAmp);
```

Outputs speaker frequency and volume with various frequency and voltage values.

#### PARAMETERS

**wFreq**—suggested frequency values are from 575 Hz to 3,000 Hz: for example, enter 1000 for 1 kHz. Values less than 575 (575 Hz) will be ignored.

**wAmp**—voltage amplitude (volume) values are 0, 1, 2, and 3: 0 = off, and 3 = loudest volume.

#### RETURN VALUE

None.

# 5. USING THE TCP/IP FEATURES

Chapter 5 provides an introduction to using the TCP/IP features on your Intellicom.

## 5.1 TCP/IP Connections

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.

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

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

- If you do not have access to an Ethernet network, use a crossover Ethernet cable to connect the Intellicom to a PC that at least has a 10Base-T Ethernet card.
- If you have an Ethernet connection, use a straight-through Ethernet cable to establish an Ethernet connection to the Intellicom from an Ethernet hub. These connections are shown in Figure 15.

3. Apply Power

Plug in the AC adapter. The Intellicom board is now ready to be used.

> **NOTE:** A hardware RESET is accomplished by unplugging the AC adapter, then plugging it back in.

When working with the Intellicom board, the green **LNK** light is on when a program is running and the board is properly connected either to an Ethernet hub or to an active Ethernet card. The red **ACT** light flashes each time a packet is received.

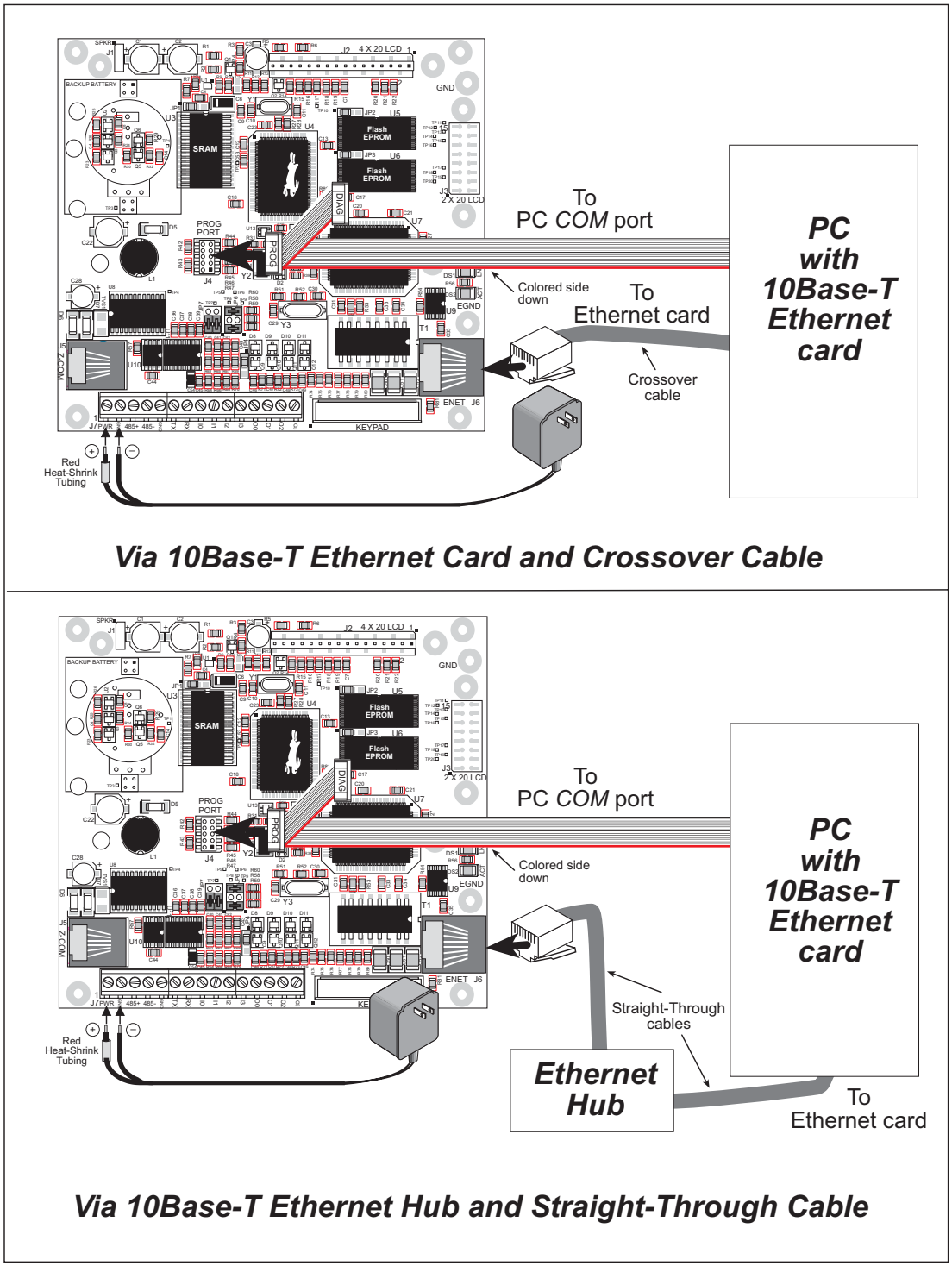



***Figure 15. Ethernet Connections***

## 5.2 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 that the user connect his PC and the Intellicom 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.

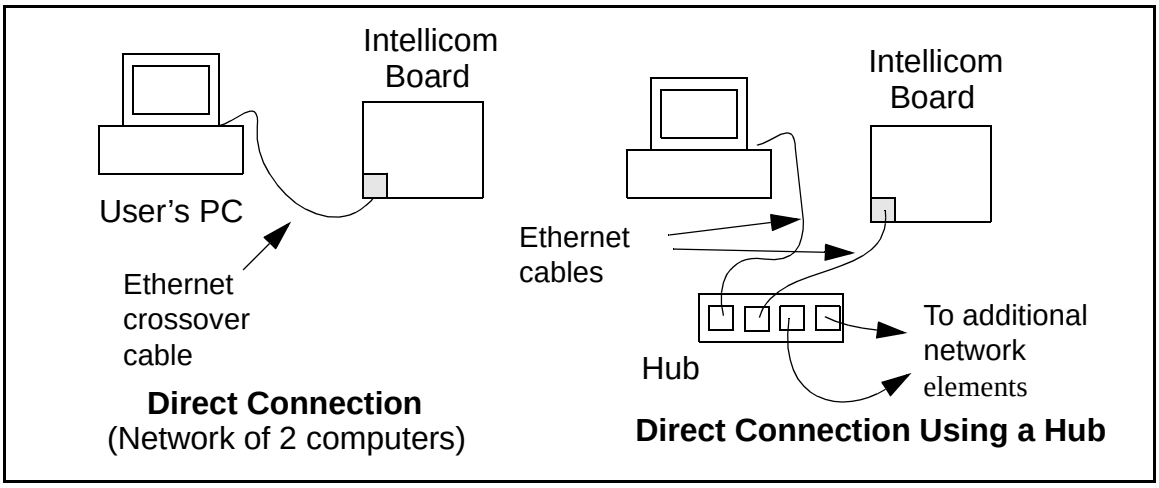
Obtaining IP addresses to interact over an existing, operating, network can involve a number of complications, and must usually be done with cooperation from your ISP and/or network systems administrator (if your company has one). For this reason, it is suggested that the user begin instead by using a direct connection between a PC and the Intellicom board using an Ethernet crossover cable or a simple arrangement with a hub. (A crossover cable should not be confused with regular straight through cables.) The hub and a wide variety of cables can also be purchased from a local computer store.

In order to set up this direct connection, the user will have to use a new PC (right out of the box), or disconnect a PC from the corporate network, or as yet another approach install a second Ethernet adapter and set up a separate private network attached to the second Ethernet adapter. Disconnecting your PC from the corporate network may be easy or nearly impossible, depending on how it is set up. Mobile PCs, such as laptops, are designed to be connected and disconnected, and will present the least problem. If your PC boots from the network or is dependent on the network for some or all of its disks, then it probably should not be disconnected. If a second Ethernet adapter is used, be aware that Windows TCP/IP will send messages to one adapter or the other, depending on the IP address and the binding order in Microstate products. Thus you should have different ranges of IP addresses on your private network from those used on the corporate network. If both networks service the same IP address, then Windows may send a packet intended for your private network to the corporate network. A similar situation will take place if you use a dial-up line to send a packet to the Internet. Windows may try to send it via the local Ethernet network if it is also valid for that network.

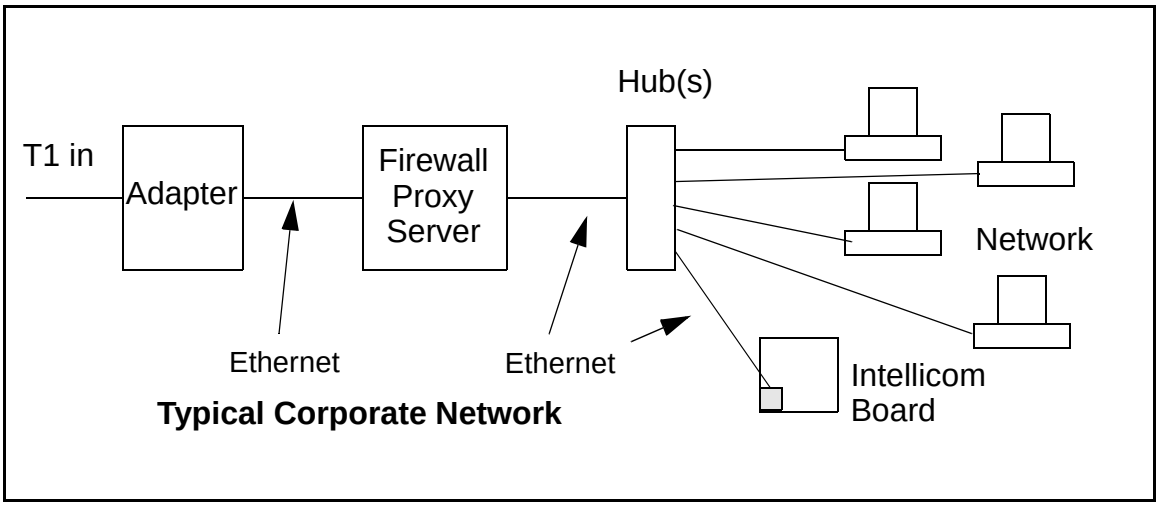
The following private IP addresses are set aside for local networks and are not allowed on the Internet: 10.0.0.0 to 10.255.255.255, 172.16.0.0 to 172.31.255.255, and 192.168.0.0 to 192.168.255.255.

The Intellicom board uses a 10Base-T type of Ethernet connection, which is the most common scheme. The RJ-45 connectors are similar to U.S. style telephone connectors, are except larger and have 8 contacts.

An alternative to the direct connection using a crossover cable is a direct connection using a hub. The hub relays packets received on any port to all of the ports on the hub. Hubs are low in cost and are readily available. The Intellicom board uses 10 Mbps Ethernet, so the hub or Ethernet adapter must be either a 10 Mbps unit or a 10/100 Mbps unit.




In a corporate setting where the Internet is brought in via a high-speed line, there are typically machines between the outside Internet and the internal network. These machines include a combination of proxy servers and firewalls that filter and multiplex Internet traffic. In the configuration below, the Intellicom board could be given a fixed address so any of the computers on the local network would be able to contact it. It may be possible to configure the firewall or proxy server to allow hosts on the Internet to directly contact the controller, but it would probably be easier to place the controller directly on the external network outside of the firewall. This avoids some of the configuration complications by sacrificing some security.




If your system administrator can give you an Ethernet cable along with its IP address, the netmask and the gateway address, then you may be able to run the sample programs without having to setup a direct connection between your computer and the Intellicom board. You will also need the IP address of the nameserver, the name or IP address of your mail server, and your domain name for some of the sample programs.

## 5.3 How to Set IP Addresses in the Sample Programs

With the introduction of Dynamic C 7.30 we have taken steps to make it easier to run many of our sample programs. You will see a **TCPCONFIG** macro. This macro tells Dynamic C to select your configuration from a list of default configurations. You will have three choices when you encounter a sample program with the **TCPCONFIG** macro.

1. You can replace the **TCPCONFIG** macro with individual **MY_IP_ADDRESS**, **MY_NETMASK**, **MY_GATEWAY**, and **MY_NAMESERVER** macros in each program.
2. You can leave **TCPCONFIG** at the usual default of 1, which will set the IP configurations to **10.10.6.100**, the netmask to **255.255.255.0**, and the nameserver and gateway to **10.10.6.1**. If you would like to change the default values, for example, to use an IP address of **10.1.1.2** for the Intellicom board, and **10.1.1.1** for your PC, you can edit the values in the section that directly follows the "General Configuration" comment in the **TCP_CONFIG.LIB** library. You will find this library in the **LIB\TCPIP** directory.
3. You can create a **CUSTOM_CONFIG.LIB** library and use a **TCPCONFIG** value greater than 100. Instructions for doing this are at the beginning of the **TCP_CONFIG.LIB** library in the **LIB\TCPIP** directory.

There are some other "standard" configurations for **TCPCONFIG** that let you select different features such as DHCP. Their values are documented at the top of the **TCP_CONFIG.LIB** library in the **LIB\TCPIP** directory. More information is available in the ***Dynamic C TCP/IP User's Manual***.

### IP Addresses Before Dynamic C 7.30

Most of the sample programs use macros to define the IP address assigned to the board and the IP address of the gateway, if there is a gateway. Instead of the **TCPCONFIG** macro, you will see a **MY_IP_ADDRESS** macro and other macros.

```
#define MY_IP_ADDRESS "10.10.6.170"
#define MY_NETMASK "255.255.255.0"
#define MY_GATEWAY "10.10.6.1"
#define MY_NAMESERVER "10.10.6.1"
```

In order to do a direct connection, the following IP addresses can be used for the Intellicom:

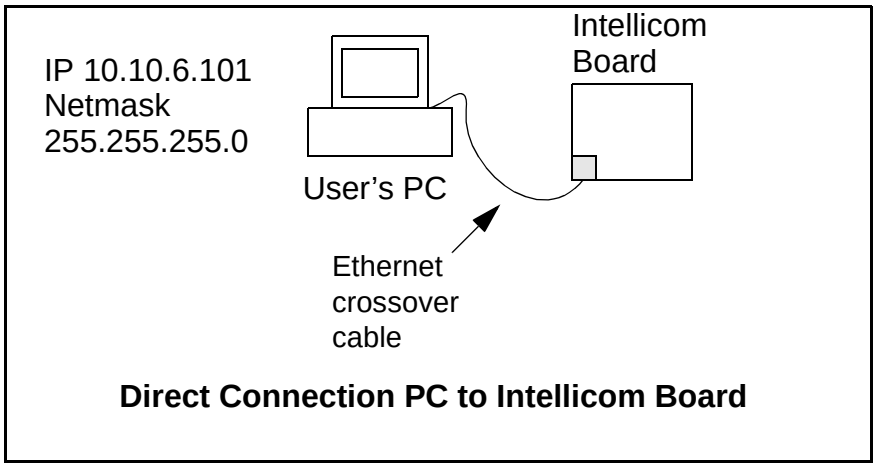
```
#define MY_IP_ADDRESS "10.1.1.2"
#define MY_NETMASK "255.255.255.0"
// #define MY_GATEWAY "10.10.6.1"
// #define MY_NAMESERVER "10.10.6.1"
```

In this case, the gateway and nameserver are not used, and are commented out. The IP address of the board is defined to be **10.1.1.2**. The IP address of you PC can be defined as **10.1.1.1**.

## 5.4 How to Set Up Your Computer's IP Address For Direct Connect

When your computer is connected directly to the Intellicom board via an Ethernet connection, you need to assign an IP address to your computer. To assign the PC the address `10.10.6.101` with the netmask `255.255.255.0`, do the following.

Click on **Start > Settings > Control Panel** to bring up the Control Panel, and then double-click the Network icon. Depending on which version of Windows you are using, look for the **TCP/IP Protocol**/**Network > Dial-Up Connections**/**Network** line or tab. Double-click on this line or select **Properties** or **Local Area Connection > Properties** to bring up the TCP/IP properties dialog box. You can edit the IP address and the subnet mask directly. (Disable "obtain an IP address automatically.") You may want to write down the existing values in case you have to restore them later. It is not necessary to edit the gateway address since the gateway is not used with direct connect.




**Direct Connection PC to Intellicom Board**

## 5.5 Run the PINGME.C Demo

In order to run this program, edit the IP address and netmask in the **PINGME.C** program (**SAMPLES\TCPIP\ICMP**) to the values given above (10.1.1.2 and 255.255.255.248). Compile the program and start it running under Dynamic C. The crossover cable is connected from your computer's Ethernet adapter to the Intellicom board's RJ-45 Ethernet connector. When the program starts running, the green LNK light on the Intellicom board should be on to indicate an Ethernet connection is made. (Note: If the LNK light does not light, you may not have a crossover cable, or if you are using a hub perhaps the power is off on the hub.)

The next step is to ping the board from your PC. This can be done by bringing up the MS-DOS window and running the pingme program:

```
ping 10.1.1.2
```

or by **Start > Run**

and typing the entry

```
ping 10.1.1.2
```

Notice that the red ACT light flashes on the Intellicom board while the ping is taking place, and indicates the transfer of data. The ping routine will ping the board four times and write a summary message on the screen describing the operation.

## 5.6 Running More Demo Programs With a Direct Connection

The programs **STATIC.C** and **SSI3.C** (**SAMPLES\TCPIP\HTTP**) demonstrate how to make the Intellicom board be a Web server. This program allows you to turn the LEDs on an attached Demonstration Board from the Development Kit on and off from a remote Web browser. In order to run these sample programs, edit the IP address as for the pingme program, compile the program and start it executing. Then bring up your Web browser and enter the following server address: http://10.1.1.2.

This should bring up the Web page served by the sample program.

The sample program **RXSAMPLE.C** (**SAMPLES\TELNET**) allows you to communicate with the Intellicom board using the Telnet protocol. To run this program, edit the IP address, compile the program, and start it running. Run the Telnet program on your PC (**Start > Run** `telnet 10.1.1.2`). Each character you type will be printed in Dynamic C's **STDIO** window, indicating that the board is receiving the characters typed via TCP/IP.

## 5.7 Where Do I Go From Here?

**NOTE:** If you purchased your Intellicom through a distributor or Rabbit partner, contact the distributor or partner first for technical support.

If there are any problems at this point:

- Use the Dynamic C **Help** menu to get further assistance with Dynamic C.
- Check the Rabbit Technical Bulletin Board and forums at www.rabbit.com/support/bb/ and at www.rabbit.com/forums/.
- Use the Technical Support e-mail form at www.rabbit.com/support/.

If the sample programs ran fine, you are now ready to go on.

Refer to the ***Dynamic C TCP/IP User's Manual*** to develop your own applications. ***An Introduction to TCP/IP*** provides background information on TCP/IP, and is included on the CD, and is also available on our Web site.

# APPENDIX A.
# INTELLICOM SPECIFICATIONS

Appendix A provides the specifications for the Intellicom and describes the conformal coating.

## A.1 Electrical and Mechanical Specifications

Figure A-1 shows the mechanical dimensions for the Intellicom board.

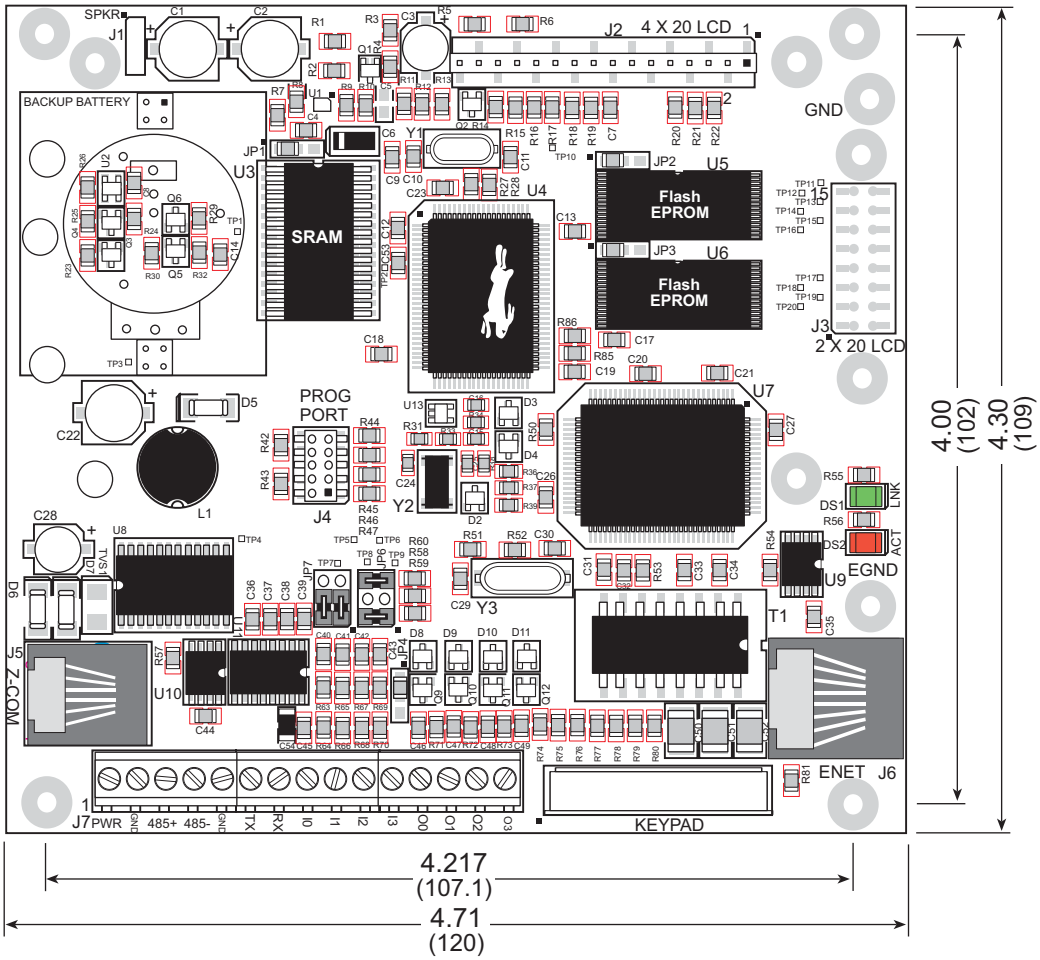



***Figure A-1. Intellicom Board Dimensions***

Table A-1 lists the electrical, mechanical, and environmental specifications for the Intellicom board.

***Table A-1. Intellicom Board Specifications***

| Feature | OP6600 | OP6700 |
|---|---|---|
| Microprocessor | Rabbit 2000™ at 18.432 MHz | |
| Ethernet Port | None | 10Base-T, RJ-45 |
| Flash EPROM | 256K | 512K (2×256K) |
| SRAM | 128K (standard) | |
| Backup Battery | On backup battery board, 3 V lithium coint-type, 1000 mA·h, supports RTC and SRAM | |
| Keypad/Display | 2 × 6 domed tactile keypad with customizable legend, supertwist 4 × 20 LCD with backlighting | |
| Digital Inputs | 4 protected to ± 36 V DC | |
| Digital Outputs | 4 open collector, sinking (200 mA, 40 V DC max.) | |
| Speaker Output | Software-adjustable volume and frequency | |
| Serial Ports | 3 serial ports:<br>• one 5 V CMOS-compatible programming port<br>• either two RS-232 (3-wire), one RS-485 and one RS-232 (3-wire), *or* one RS-232 (5-wire) | |
| Serial Rate | Max. burst rate = CLK/32<br>Max. sustained rate = Burst/2 | |
| Connectors | 15 screw terminals, 1 RJ-12, and 1 RJ-45 | |
| Real-Time Clock | Yes | |
| Timers | 5 eight-bit timers (four cascadable from the first) and one 10-bit timer with two match registers | |
| Watchdog/Supervisor | Yes | |
| Power | 9–40 V DC, 2.4 W (backlighting on) | |
| Operating Temperature | 0°C to +50°C | |
| Storage Temperature | –20°C to +60°C | |
| Humidity | 5% to 95%, noncondensing | |
| Board Size (with backup battery board) | 4.30" × 4.71" × 0.80"<br>(109 mm × 120 mm × 20.3 mm) | |
| Enclosure Size | 5.7" × 6.7" × 2.0"<br>(145 mm × 170 mm × 51 mm) | |

## A.2 Conformal Coating

The areas around the crystal oscillator and the battery-backup circuit on the Intellicom have had the Dow Corning silicone-based 1-2620 conformal coating applied. The conformally coated areas are shown in Figure A-2. The conformal coating protects these high-impedance circuits from the effects of moisture and contaminants over time.

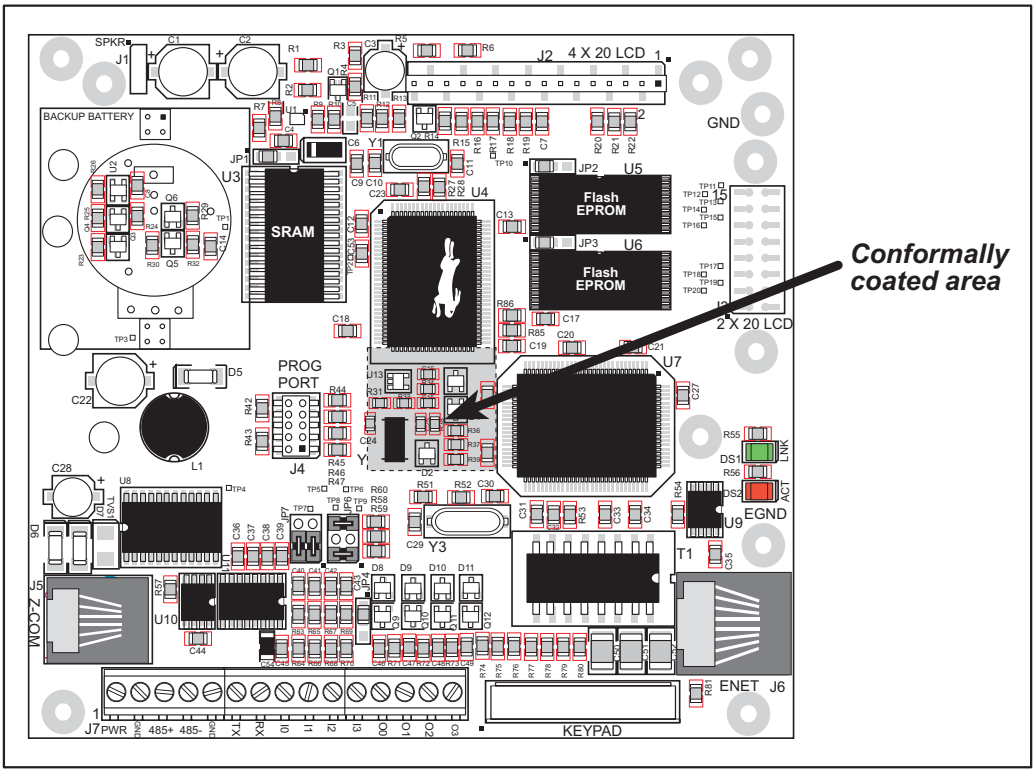



***Figure A-2. Intellicom Areas Receiving Conformal Coating***

Any components in the conformally coated area may be replaced using standard soldering procedures for surface-mounted components. A new conformal coating should then be applied to offer continuing protection against the effects of moisture and contaminants.

> **NOTE:** For more information on conformal coatings, refer to Technical Note 303, ***Conformal Coatings***.

## A.3 Jumper Configurations

Figure A-3 shows the header locations used to configure the various Intellicom options via jumpers.

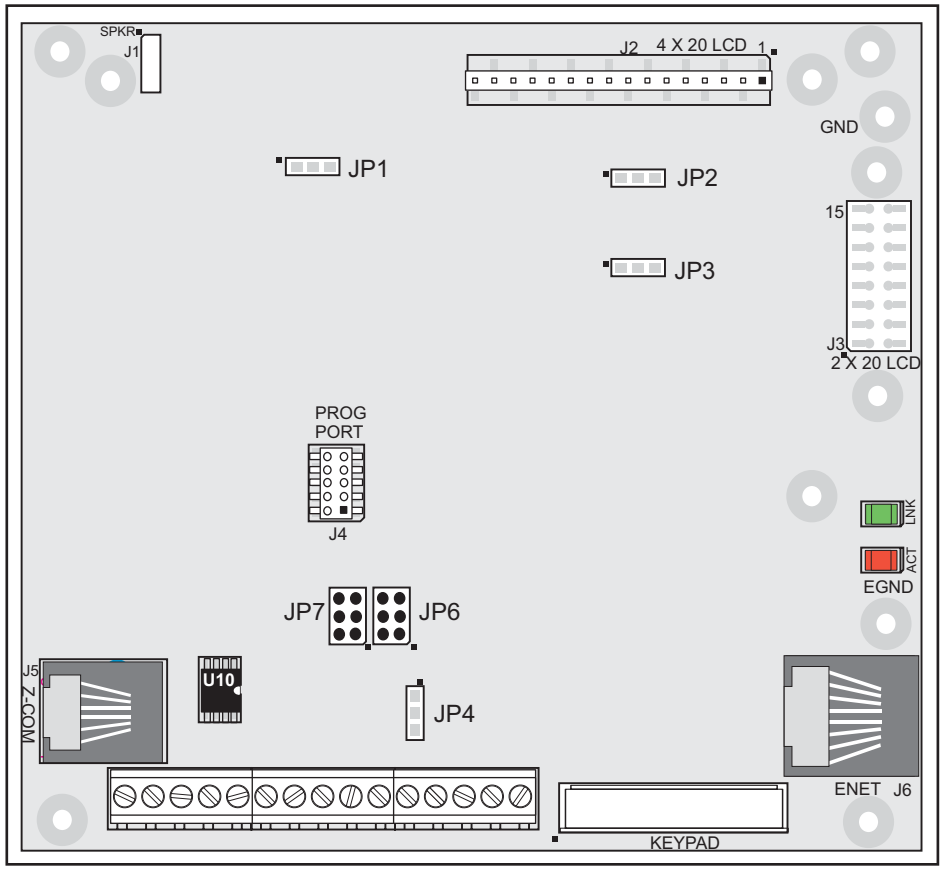



***Figure A-3. Location of Intellicom Configurable Positions***

Table A-2 lists the configuration options.

***Table A-2. Intellicom Jumper Configurations***

| Header | Description | Pins Connected | | Factory Default |
|---|---|---|---|---|
| JP1 | SRAM Size | 1–2 | 128K | ✕ |
| | | 2–3 | 512K | |
| | | None | 32K | |
| JP2 | Flash 1 Memory Size (U5) | 1–2 | 128K/256K | ✕ |
| | | 2–3 | 512K | |
| JP3 | Flash 2 Memory Size (U6) | 1–2 | 128K/256K | ✕ |
| | | 2–3 | 512K | |
| JP4 | Digital Input Pull-Up/Pull-Down Resistors | 1–2 | Pulled up | ✕ |
| | | 2–3 | Pulled down | |
| JP5 | Flash Memory Bank Select | 1–2 | Normal Mode | ✕ |
| | | 2–3 | Bank Mode | |
| JP6 | RS-485 Bias and Termination Resistors | 1–2<br>5–6 | Bias and termination resistors connected | ✕ |
| | | None | Bias and termination resistors ***not*** connected | |
| JP7 | RS-232/RS-485 Select | 1–3<br>2–4 | RS-232 TxB/RxB (also TxC/RxC or RTS/CTS) signals on J7 | |
| | | 3–5<br>4–6 | RS-232 TxC/RxC and RS-485 signals on J7 | ✕ |
| | | 1–5<br>2–6 | RS-232 TxB/RxB signals on J5 (U10 must be removed) | |

**NOTE:** Only headers JP6 and JP7 use actual jumpers. The other connections are made using 0 Ω surface-mounted resistors.

# APPENDIX B.
# KEYPAD AND PLASTIC ENCLOSURE

## B.1 Keypad Insert

The keypad is designed to accept paper inserts prepared on regular paper. The templates shown below in Figure B-1 can be used to create custom inserts. The numbers shown on the upper template correspond to the codes returned by Dynamic C when that key is pressed.

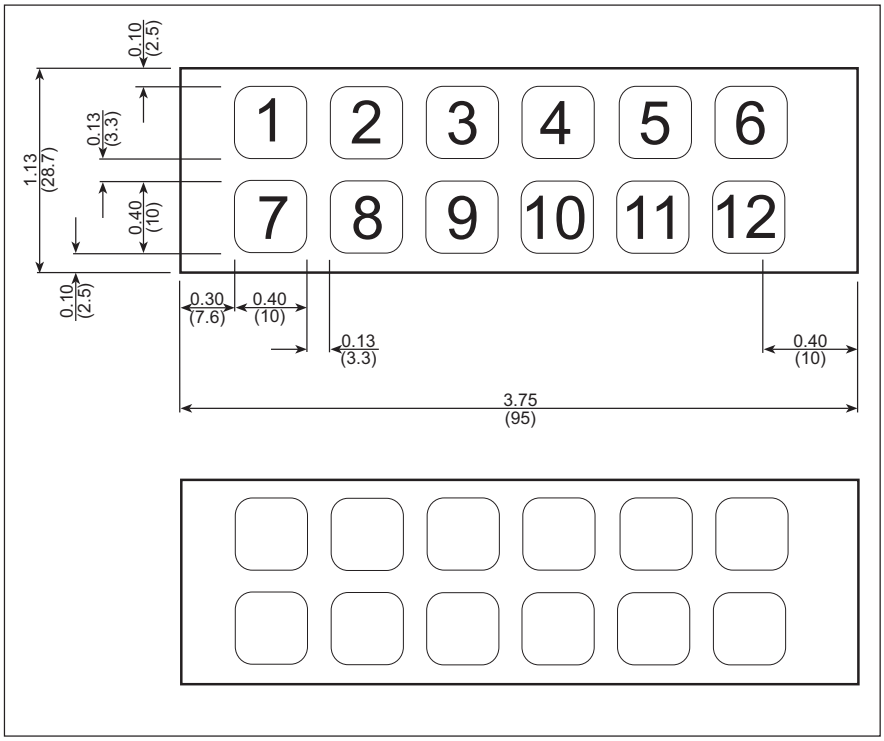



***Figure B-1. Intellicom 2 × 6 Keypad Template***

These instructions describe how to change the keypad insert.

1. Set the outer plastic casing aside and lay the front panel face down on a soft cloth so that the Intellicom board is facing up.
2. Unplug the speaker and the keypad as shown in Figure B-2.
3. Remove the four screws shown in Figure B-2 that hold the Intellicom board to the front panel.
4. Lift the Intellicom board up front the front panel and set it aside. Note that the LCD is attached permanently to the Intellicom board.
5. Remove the four screws shown in Figure B-3 that hold the keypad to the front panel.
6. Remove the old insert and place the new insert in between the keypad and the mylar front. You may tape down the portion of the insert that extends beyond the keypad.
7. Line up the keypad over the front panel and replace the four 2-56 screws as shown in Figure B-3. Line up the Intellicom board/LCD and replace the four screws as shown in Figure B-2. Reconnect the keypad and the speaker to the Intellicom board.

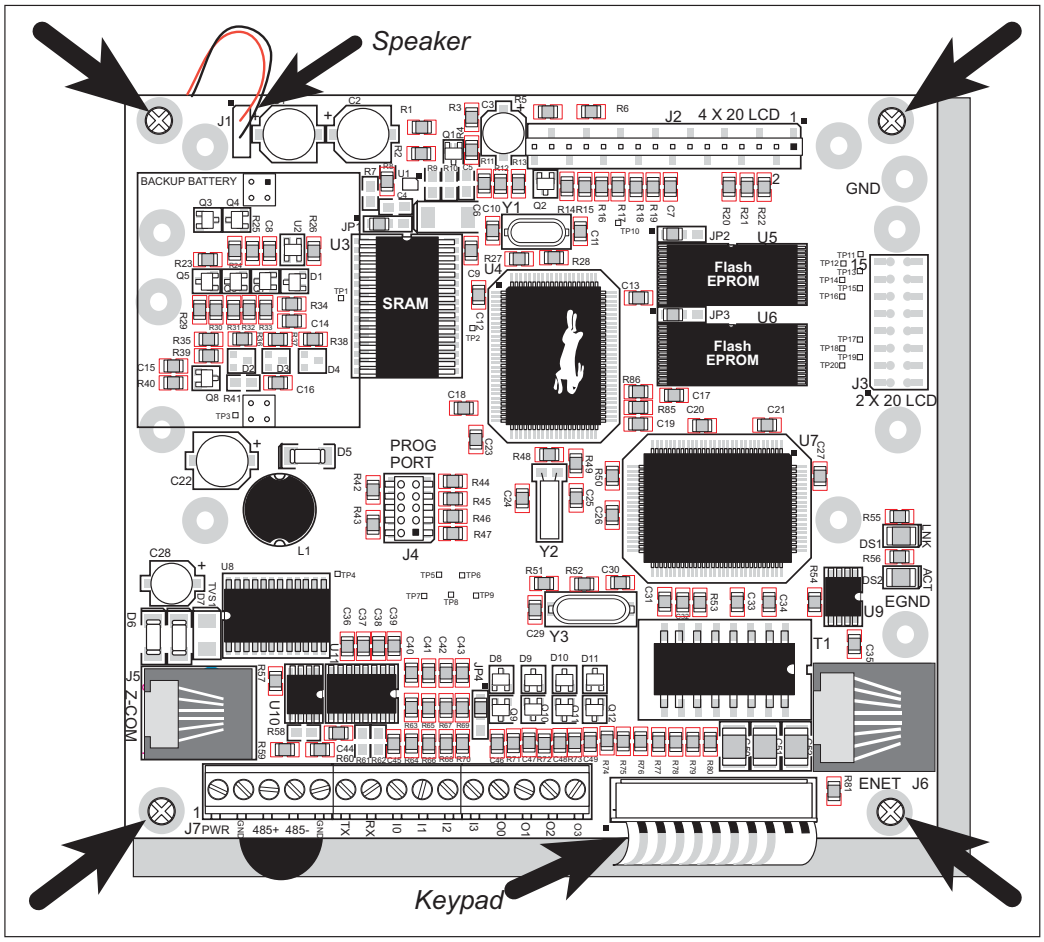



***Figure B-2. Removing Intellicom Board from Front Panel***

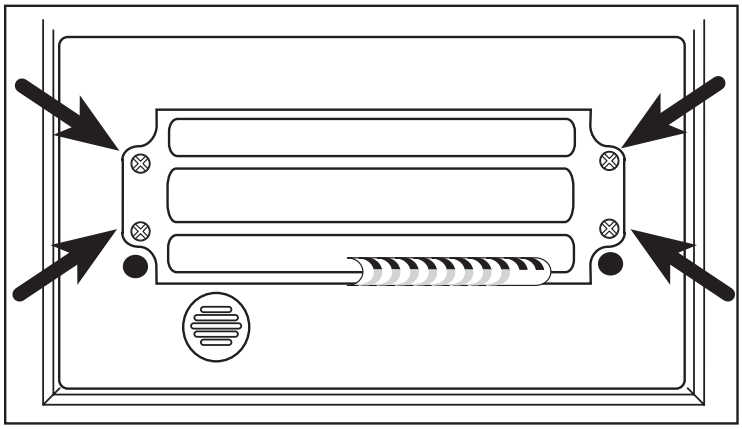


***Figure B-3. Removing Keypad from Front Panel***

## B.2 Plastic Enclosure

Figure B-4 shows the dimensions of the Intellicom front panel bezel.

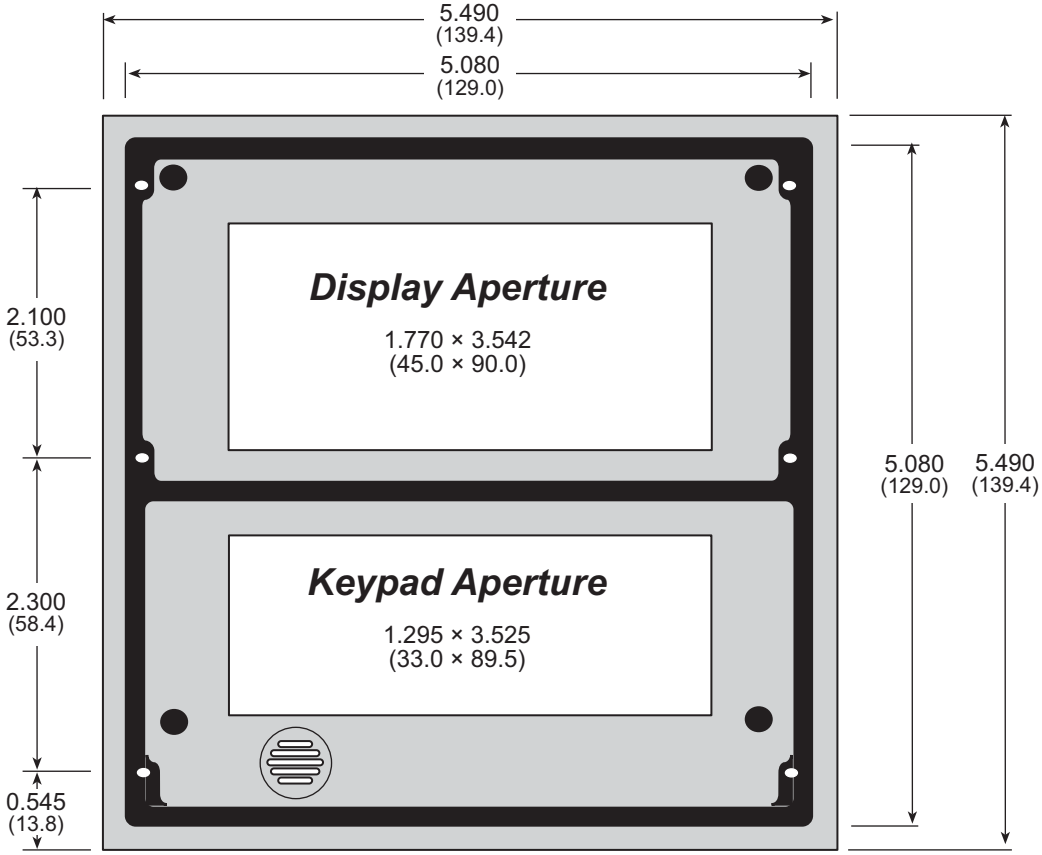



***Figure B-4. Intellicom Front Panel Bezel Dimensions***

Figure B-5 shows the dimensions of the outer casing, including the attached front panel.

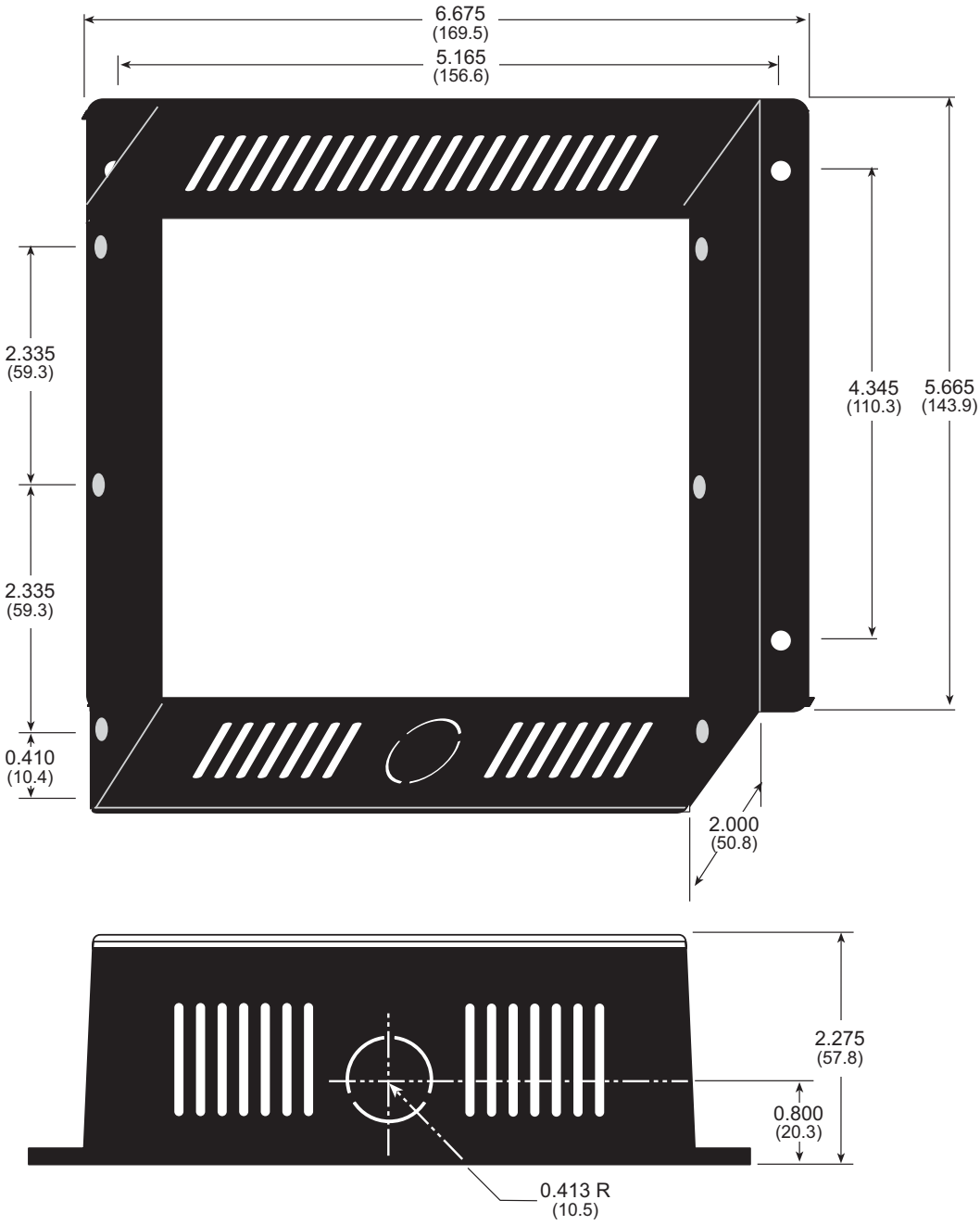



***Figure B-5. Intellicom Outer Casing Dimensions***

## B.2.1 Assembling Intellicom Enclosure

There are two recommended assemblies possible for the Intellicom:

1. Mount the front panel bezel in an opening you have created. This option allows you to have a splash-resistant unit by using the gasket supplied with the Intellicom to form a splash-resistant seal between the front panel bezel and your opening.
2. Mount the front panel bezel using the outer casing supplied with the Intellicom.

### B.2.1.1 Custom Mounting In An Opening

Prepare an opening to accommodate the Intellicom front panel bezel shown in Figure B-4. The thickness of the surface the Intellicom front bezel is mounted on should be either 0.0625 inches (1.6 mm) or 0.125 inches (3.2 mm). Use the brackets on the plastic tree included with the Intellicom (see Figure B-6) to attach the front panel bezel to the surface.

A splash-resistant mounting is also possible.

First, remove the speaker from the front panel bezel, and insert the speaker plug to close the holes in the speaker grille. Apply some hot glue around the edges of the plug to form a splash-resistant seal. Replace the speaker—a dab of glue around the edge of the speaker will hold it in place.

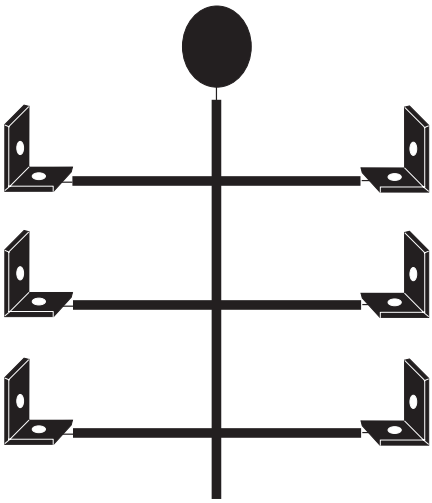


*Figure B-6. Mounting Brackets and Speaker Plug*

Place the rubber gasket supplied with the Intellicom between the front panel bezel and the surface the Intellicom will be mounted. A splash-resistant seal will result when the brackets are attached.

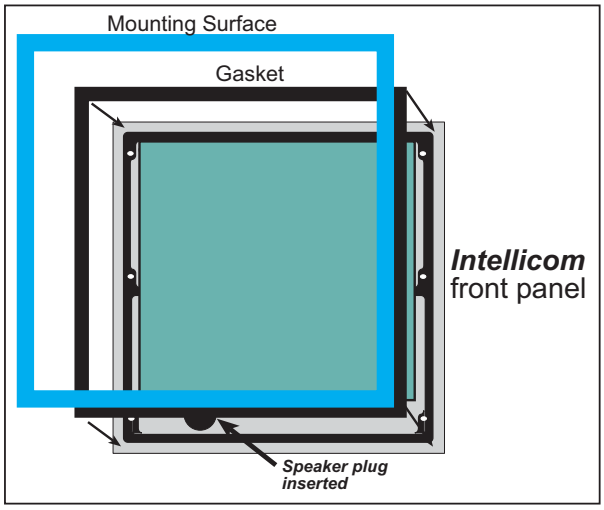



*Figure B-7. Splash-Resistant Mounting of Intellicom*

### B.2.1.2 Supplied Outer Casing

Once the desired wires have been connected to header J7 on the Intellicom board, the Intellicom may be mounted in the plastic outer casing as shown in Figure B-8. Secure the plastic casing with the six screws supplied with the Intellicom

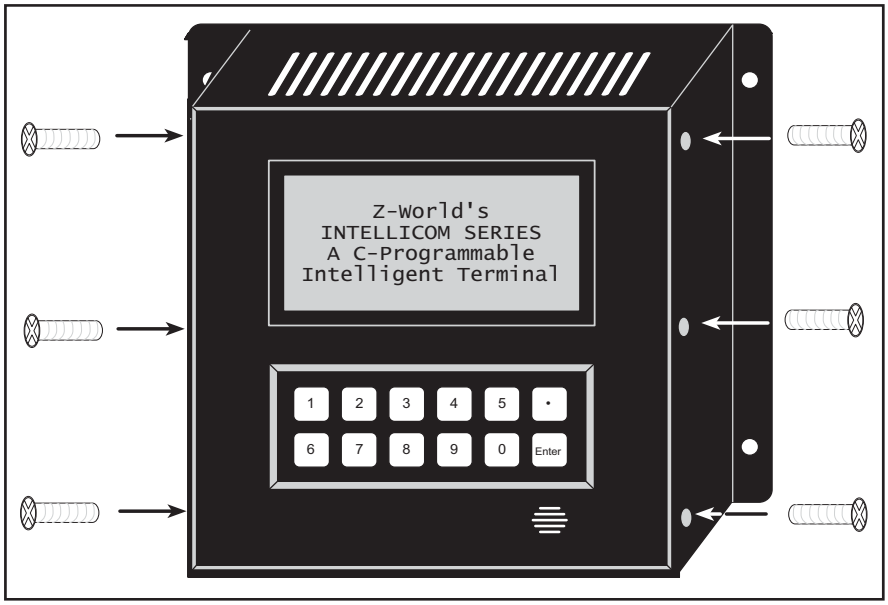



***Figure B-8. Mounting Intellicom in Plastic Outer Casing***

If you are using the RJ-45 jack at J6 for an Ethernet connection with the Intellicom installed in the plastic outer casing, remove the protective rubber boot around the Ethernet cable RJ-45 jack that plugs into the Intellicom board. This will ensure that the Ethernet cable can bend back within the depth of the plastic outer casing.

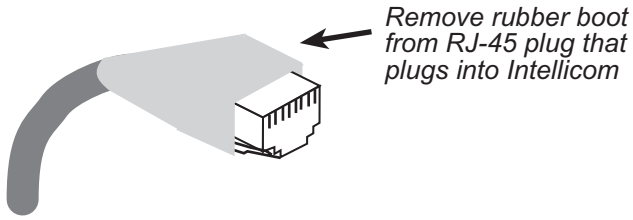



***Figure B-9. Remove Rubber Boot from RJ-45 Connection to Intellicom***

The outer casing has a knockout for a conduit on one side. The casing is symmetric, and so the conduit knockout can face up or down. Use a small hacksaw to remove the knockout if you are using a conduit. The conduit opening has an O.D. of 0.826 inches (21.0 mm), which accommodates standard trade size ½ or 17 mm diameter conduit.

When routing cables through the conduit, our has found that an Ethernet cable (minus the rubber boot on the RJ-45 plug) should be routed first, followed by the RS-485 cable with the RJ-12 plug, followed by other wire.

Figure B-10 shows an Intellicom wired through a conduit.

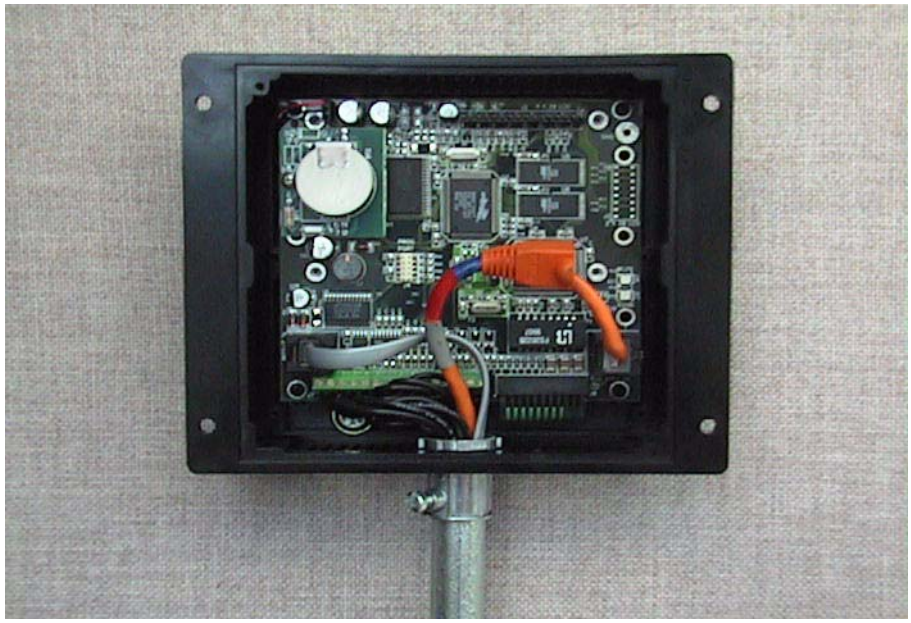


***Figure B-10. Rear View of Intellicom with Outer Casing Showing Wiring from Conduit***

# APPENDIX C. POWER MANAGEMENT

Appendix C describes the power circuitry distributed on the Intellicom.

## C.1 Power Supplies

Power is supplied to the Intellicom board from an external source either through header J7 or from another Intellicom through header J5, the RJ-12 jack.

The Intellicom board itself is protected against reverse polarity by Shottky diodes at D6 and D7 as shown in Figure C-1. The Shottky diode has a low forward voltage drop, 0.3 V, which keeps the minimum DCIN required to power the Intellicom lower than a normal silicon diode would allow.

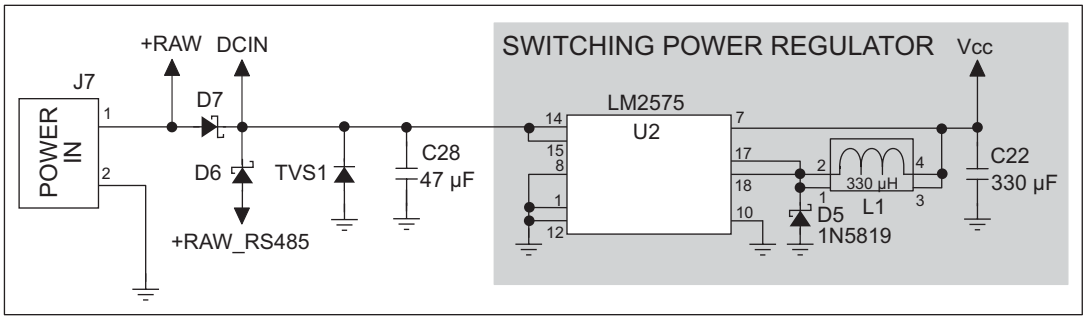



***Figure C-1. Intellicom Power Supply Schematic***

Capacitor C28 provides surge current protection for the voltage regulator, and allows the external power supply to be located some distance away from the Intellicom board. A switching power regulator is used. The input voltage range is from 9 V to 40 V.

## C.2 Batteries and External Battery Connections

A battery board with a 1000 mA·h lithium coin cell provides power to the real-time clock and SRAM when external power is removed from the circuit. This allows the Intellicom to continue to keep track of time and preserves the SRAM memory contents.

Figure C-2 shows the battery-board circuit.

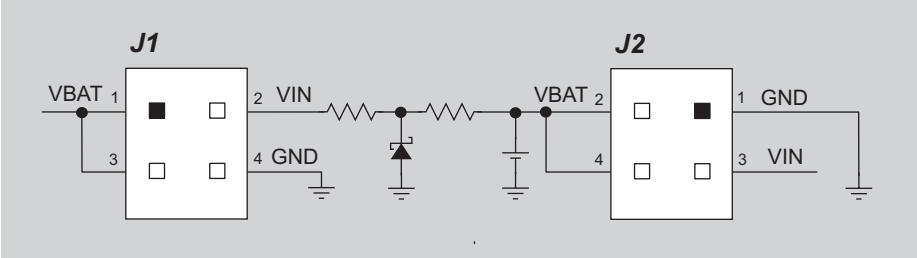



***Figure C-2. Intellicom Backup-Battery Board***

Alternatively, starting with the 175-0206 version of the Intellicom, there is provision to add a soldered-in battery directly on the Intellicom board.

The drain on the battery is typically less than 20 µA when there is no external power applied. The battery can last more than 5 years:

$$\frac{1000\ \text{mA}\cdot\text{h}}{20\ \mu\text{A}} = 5.7\ \text{years}.$$

The drain on the battery is typically less than 4 µA when external power ***is*** applied. The battery can last for its full shelf life:

$$\frac{1000\ \text{mA}\cdot\text{h}}{4\ \mu\text{A}} = 28.5\ \text{years (shelf life = 10 years)}.$$

Since the shelf life of the battery is 10 years, the battery can last for its full shelf life when external power is applied to the Intellicom.

### C.2.1 Battery-Backup Circuit

The battery-backup circuit serves two purposes:

- It reduces the battery voltage to the real-time clock, thereby reducing the current consumed by the real-time clock and lengthening the battery life.
- It ensures that current can flow only ***out*** of the battery to prevent charging the battery.

Figure C-3 shows the battery-backup circuitry on the Intellicom board.

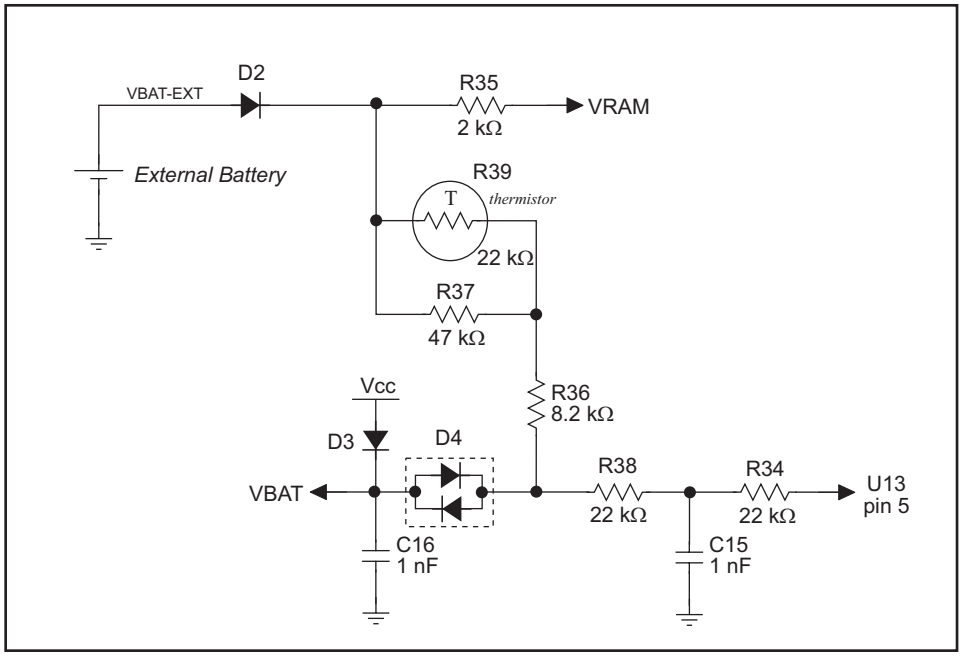



***Figure C-3. Intellicom Battery-Backup Circuit***

The battery-backup circuit serves three purposes:

- It reduces the battery voltage to the SRAM and to the real-time clock, thereby limiting the current consumed by the real-time clock and lengthening the battery life.
- It ensures that current can flow only ***out*** of the battery to prevent charging the battery.
- A voltage, VOSC, is supplied to U13, which keeps the 32.768 kHz oscillator working when the voltage begins to drop.

VRAM and Vcc are nearly equal (<100 mV, typically 10 mV) when power is supplied to the Intellicom. VRAM is also available on pin 34 of header J2 to facilitate battery backup of the external circuit. Note that the recommended minimum resistive load at VRAM is 100 kΩ, and new battery life calculations should be done to take external loading into account.

### C.2.2 Power to VRAM Switch

The VRAM switch, shown in Figure C-4, allows the battery backup to provide power when the external power goes off. The switch provides an isolation between Vcc and the battery when Vcc goes low. This prevents the Vcc line from draining the battery.

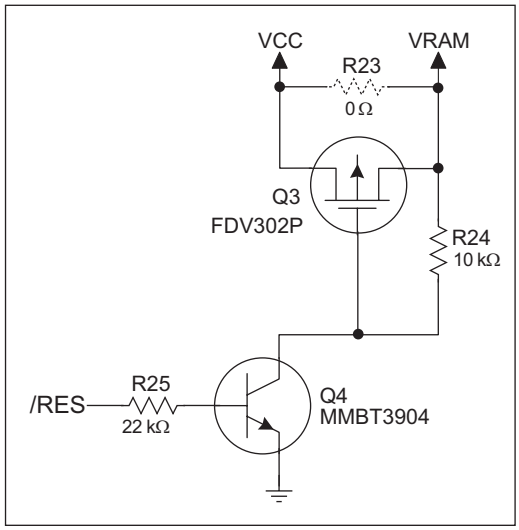



***Figure C-4. VRAM Switch***

Transistor Q3 is needed to provide a very small voltage drop between Vcc and VRAM (<100 mV, typically 10 mV) so that the processor lines powered by Vcc will not have a significantly different voltage than VRAM.

When the Intellicom is ***not*** resetting (pin 2 on U4 is high), the /RES line will be high. This turns on Q4, causing its collector to go low. This turns on Q3, allowing VRAM to nearly equal Vcc.

When the Intellicom ***is*** resetting, the /RES line will go low. This turns off Q3 and Q4, providing an isolation between Vcc and VRAM.

The battery-backup circuit keeps VRAM from dropping below 2 V.

### C.2.3 Reset Generator

The Intellicom uses a reset generator, U2, to reset the Rabbit 2000 microprocessor when the voltage drops below the voltage necessary for reliable operation. The reset occurs between 4.50 V and 4.75 V, typically 4.63 V.

### C.2.4 Replacing the Backup-Battery Board

The pluggable backup-battery board makes it easy to replace the backup battery with a fresh backup battery on another backup battery board. Before replacing the backup-battery board, make sure that the Intellicom is receiving power from the standard power supply. This makes sure that data in RAM are not lost when the battery-backup board is removed temporarily.

To replace the backup-battery board, remove the screw and unplug the old battery board as shown in Figure C-5.

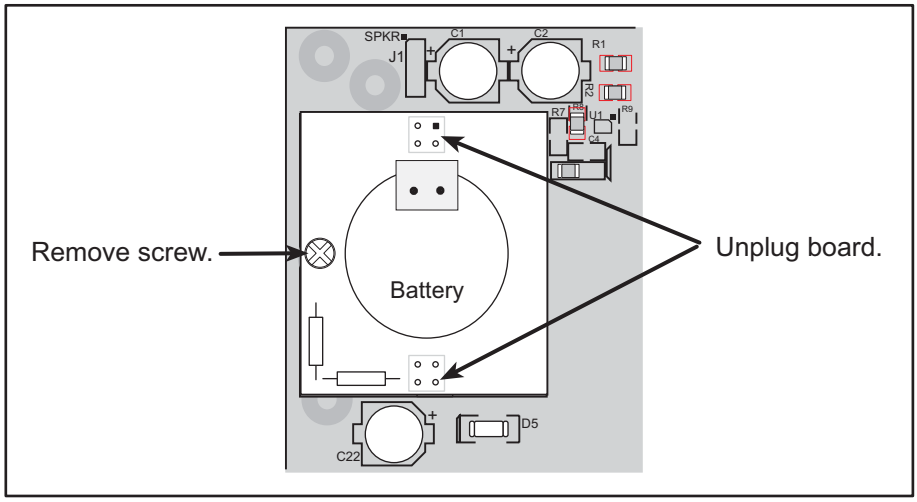



***Figure C-5. Replacing Backup-Battery Board***

Then align the replacement battery board over the outline, and plug it in. Be careful to align the connectors. Replace the screw.

Do ***not*** attempt to recharge the old battery and do ***not*** dispose of it in regular trash to avoid any risk of explosion or fire. You may either return the old backup-battery board to Rabbit for recycling or send the battery yourself to an approved recycling facility.

## C.3 Chip Select Circuit

Figure C-6 shows a schematic of the chip select circuit.

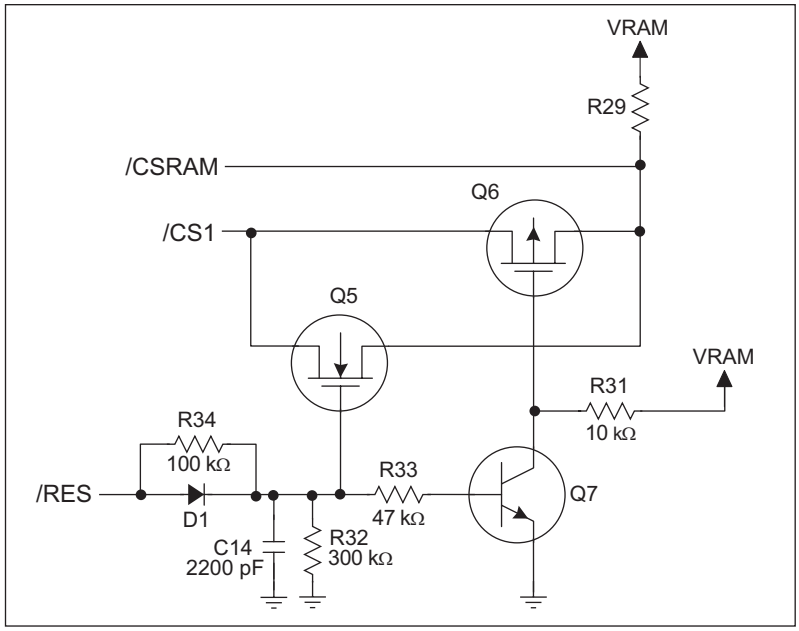



***Figure C-6. Chip Select Circuit***

The current drain on the battery in a battery-backed circuit must be kept at a minimum. When the Intellicom is not powered, the battery keeps the SRAM memory contents and the real-time clock (RTC) going. The SRAM has a powerdown mode that greatly reduces power consumption. This powerdown mode is activated by raising the chip select (CS) signal line. Normally the SRAM requires Vcc to operate. However, only 2 V is required for data retention in powerdown mode. Thus, when power is removed from the circuit, the battery voltage needs to be provided to both the SRAM power pin and to the CS signal line. The CS control circuit accomplishes this task for the CS signal line.

In a powered-up condition, the CS control circuit must allow the processor's chip select signal /CS1 to control the SRAM's CS signal /CSRAM. So, with power applied, /CSRAM must be the same signal as /CS1, and with power removed, /CSRAM must be held high (but only needs to be battery voltage high). Q5 and Q6 are MOSFET transistors with opposing polarity. They are both turned on when power is applied to the circuit. They allow the CS signal to pass from the processor to the SRAM so that the processor can periodically access the SRAM. When power is removed from the circuit, the transistors will turn off and isolate /CSRAM from the processor. The isolated /CSRAM line has a 100 kΩ pullup resistor to VRAM (R29). This pullup resistor keeps /CSRAM at the VRAM voltage level (which under no power condition is the backup battery's regulated voltage at a little more than 2 V).

Transistors Q5 and Q6 are of opposite polarity so that a rail-to-rail voltage can be passed. When the /CS1 voltage is low, Q5 will conduct. When the /CS1 voltage is high, Q6 will conduct. It takes time for the transistors to turn on, creating a propagation delay. This delay is typically very small, about 10 ns to 15 ns.

The signal that turns the transistors on is a high on the processor's reset line, /RES. When the Intellicom is not in reset, the reset line will be high, turning on n-channel Q5 and Q7. Q7 is a simple inverter needed to turn on Q6, a p-channel MOSFET. When a reset occurs, the /RES line will go low. This will cause C14 to discharge through R32 and R34. This small delay (about 160 µs) ensures that there is adequate time for the processor to write any last byte pending to the SRAM before the processor puts itself into a reset state. When coming out of reset, CS will be enabled very quickly because D1 conducts to charge capacitor C14.

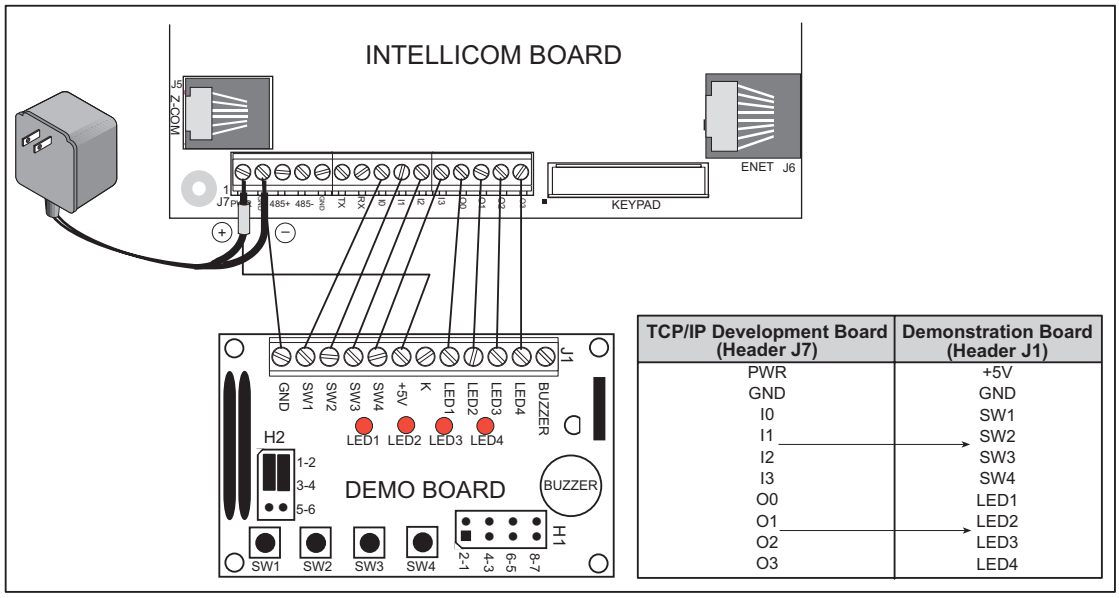
# APPENDIX D. RUNNING SAMPLE PROGRAMS

Appendix D shows how to connect the Demonstration Board to the Intellicom board, and goes through a detailed look at one sample program and the associated features in Dynamic C.

## D.1 Connecting Demonstration Board

Before running sample programs based on the Demonstration Board, you will have to connect the Demonstration Board from the Intellicom Development Kit to the Intellicom board. Proceed as follows.

1. Use the wires included in the Intellicom Tool Kit to connect header J1 on the Demonstration Board to header J7 on the Intellicom board. The connections are shown in Figure D-1.
2. Make sure that your Intellicom board is connected to your PC and that the power supply is connected to the Intellicom board and plugged in as described in Chapter 2, "Getting Started,"



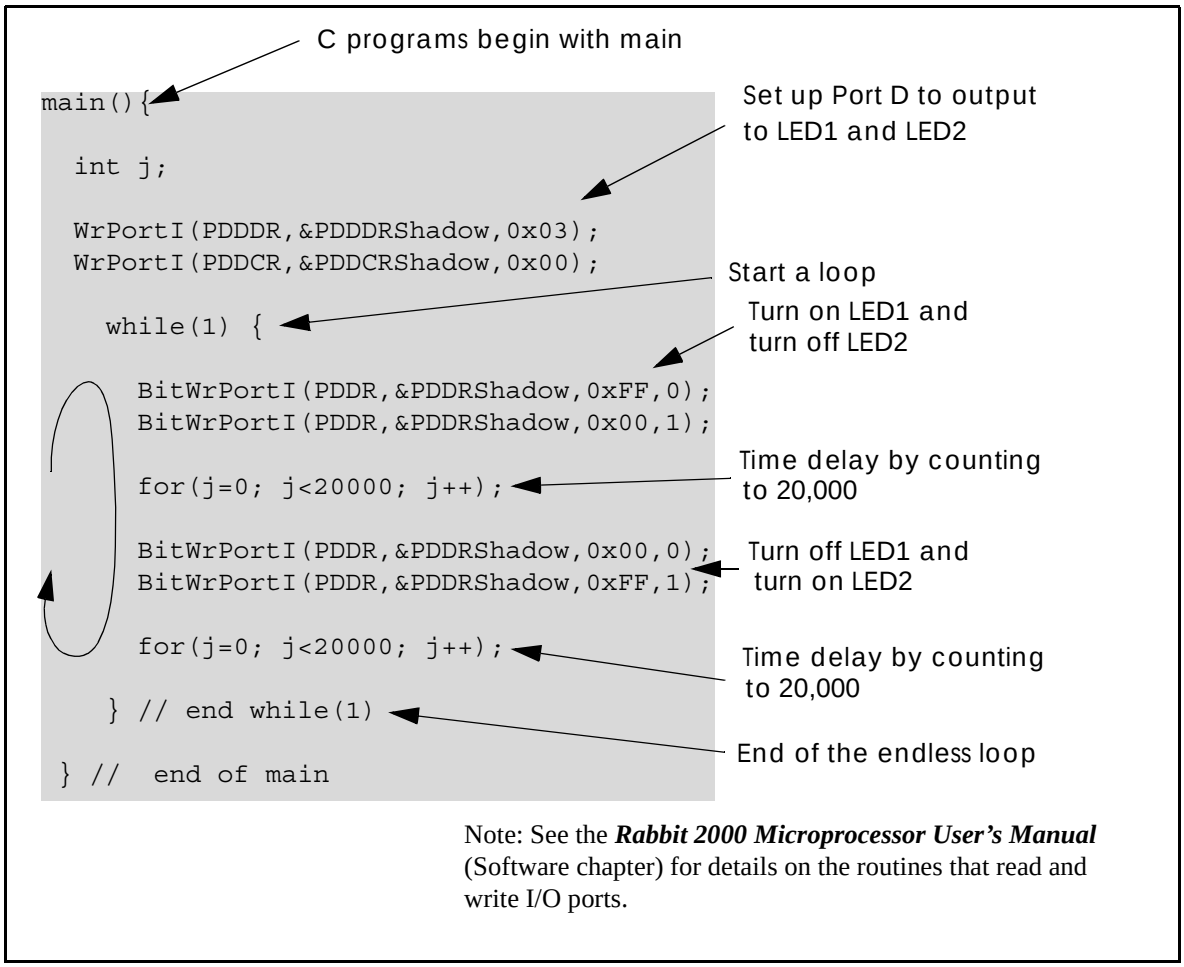

| TCP/IP Development Board (Header J7) | Demonstration Board (Header J1) |
|---|---|
| PWR | +5V |
| GND | GND |
| I0 | SW1 |
| I1 | SW2 |
| I2 | SW3 |
| I3 | SW4 |
| O0 | LED1 |
| O1 | LED2 |
| O2 | LED3 |
| O3 | LED4 |

***Figure D-1. Connections Between Intellicom Board and Demonstration Board***

## D.2 Running Sample Program DEMOBRD1.C

The sample program **DEMOBRD1.C** in the **SAMPLES\ICOM** folder will be used to illustrate some of the functions of Dynamic C.

Now, open **DEMOBRD1.C**. The program will appear in a window, as shown in Figure D-2 below (minus some comments). Use the mouse to place the cursor on the function name **WrPortI** in the program and type **<ctrl-H>**. This will bring up a documentation box for the function **WrPortI**. In general, you can do this with all functions in Dynamic C libraries, including libraries you write yourself. Close the documentation box and continue.




Note: See the ***Rabbit 2000 Microprocessor User's Manual*** (Software chapter) for details on the routines that read and write I/O ports.

***Figure D-2. Sample Program DEMOBRD1.C***

To run the program **DEMOBRD1.C**, open it with the **File** menu, compile it using the **Compile** menu, and then run it by selecting **Run** in the **Run** menu. LED1 and LED2 on the Demonstration Board should start going on and off if everything went well. If this doesn't work, review the following points.

- The target should be ready, which is indicated by the message "BIOS successfully compiled..." If you did not receive this message or you get a communication error, recompile the BIOS by typing **<ctrl-Y>** or select **Recompile BIOS** from the **Compile** menu.

- A message reports “No Rabbit Processor Detected” in cases where the wall transformer is either not connected or is not plugged in.
- The programming cable must be connected to the Intellicom board. (The colored wire on the programming cable is closest to pin 1 on header J4 on the Intellicom board, as shown in Figure 1.) The other end of the programming cable must be connected to the PC serial port. The COM port specified in the Dynamic C **Options** menu must be the same as the one the programming cable is connected to.
- To check if you have the correct serial port, select **Compile**, then **Compile BIOS**, or type **<ctrl-Y>**. If the “BIOS successfully compiled …” message does not display, try a different serial port using the Dynamic C **Options** menu until you find the serial port you are plugged into. Don’t change anything in this menu except the COM number. The baud rate should be 115,200 bps and the stop bits should be 1.

### D.2.1 Single-Stepping

Compile or re-compile **DEMOBRD1.C** by clicking the **Compile** button on the task bar. The program will compile and the screen will come up with a highlighted character (green) at the first executable statement of the program. Use the **F8** key to single-step. Each time the **F8** key is pressed, the cursor will advance one statement. When you get to the **for(j=0, j< ...** statement, it becomes impractical to single-step further because you would have to press **F8** thousands of times. We will use this statement to illustrate watch expressions.

#### D.2.1.1 Watch Expression

Type **<ctrl-W>** or chose **Add/Del Watch Expression** in the **Inspect** menu. A box will come up. Type the lower case letter j and click on *add to top* and *close*. Now continue single-stepping with **F8**. Each time you step, the watch expression (**j**) will be evaluated and printed in the watch window. Note how the value of **j** advances when the statement **j++** is executed.

#### D.2.1.2 Break Point

Move the cursor to the start of the statement:

```
for(j=0; j<20000; j++);
```

To set a break point on this statement, type **F2** or select **Breakpoint** from the **Run** menu. A red highlight will appear on the first character of the statement. To get the program running at full speed, type **F9** or select **Run** on the **Run** menu. The program will advance until it hits the break point. The break point will start flashing both red and green colors. Note that LED1 on the Demonstration Board is now solidly turned on. This is because we have passed the statement turning on LED1.

To remove the break point, type **F2** or select **Toggle Breakpoint** on the **Run** menu. To continue program execution, type **F9** or select **Run** from the **Run** menu. Now LED1 should be flashing again because the program is running at full speed.

You can set break points while the program is running by positioning the cursor to a statement and using the **F2** key. If the execution thread hits the break point, a break point will

take place. You can toggle the break point off with the **F2** key and continue execution with the **F9** key. Try this a few times to get the feel of things.

#### D.2.1.3 Editing the Program

Click on the **Edit** box on the task bar. This will set Dynamic C into the edit mode so that you can change the program. Use the **Save as** choice on the **File** menu to save the file with a new name so as not to change the demo program. Save the file as **MYTEST.C**. Now change the number 20000 in the **for (..** statement to 10000. Then use the **F9** key to recompile and run the program. The LEDs will start flashing, but it will flash much faster than before because you have changed the loop counter terminal value from 20000 to 10000.

#### D.2.1.4 Watching Variables Dynamically

Go back to edit mode (select edit) and load the program **DEMOBRD2.C** using the **File** menu **Open** command. This program is the same as the first program, except that a variable **k** has been added along with a statement to increment **k** each time around the endless loop. The statement:

```
runwatch();
```

has been added. This is a debugging statement that makes it possible to view variables while the program is running.

Use the **F9** key to compile and run **DEMOBRD2.C**. Now type **<ctrl-W>** to open the watch window and add the watch expression **k** to the top of the list of watch expressions. Now type **<ctrl-U>**. Each time you type **<ctrl-U>**, you will see the current value of **k**, which is incrementing about 5 times a second.

As an experiment add another expression to the watch window:

```
k*5
```

Then type **<ctrl-U>** several times to observe the watch expressions **k** and **k*5**.

#### D.2.1.5 Summary of Features

So far you have practiced using the following features of Dynamic C.

- Loading, compiling and running a program. When you load a program it appears in an edit window. You can compile by selecting **Compile** on the task bar or from the **Compile** menu. When you compile the program, it is compiled into machine language and downloaded to the target over the serial port. The execution proceeds to the first statement of main where it pauses, waiting for you to command the program to run, which you can do with the **F9** key or by selecting **Run** on the **Run** menu. If want to compile and start the program running with one keystroke, use **F9**, the run command. If the program is not already compiled, the run command will compile it first.
- Single-stepping. This is done with the **F8** key. The **F7** key can also be used for single-stepping. If the **F7** key is used, then descent into subroutines will take place. With the **F8** key the subroutine is executed at full speed when the statement that calls it is stepped over.

- Setting break points. The **F2** key is used to turn on or turn off (toggle) a break point at the cursor position if the program has already been compiled. You can set a break point if the program is paused at a break point. You can also set a break point in a program that is running at full speed. This will cause the program to break if the execution thread hits your break point.
- Watch expressions. A watch expression is a C expression that is evaluated on command in the watch window. An expression is basically any type of C formula that can include operators, variables and function calls, but not statements that require multiple lines such as *for* or *switch*. You can have a list of watch expressions in the watch window. If you are single-stepping, then they are all evaluated on each step. You can also command the watch expression to be evaluated by using the **<ctrl-U>** command. When a watch expression is evaluated at a break point, it is evaluated as if the statement was at the beginning of the function where you are single-stepping. If your program is running you can also evaluate watch expressions with a **<ctrl-U>** if your program has a **`runwatch()`** command that is frequently executed. In this case, only expressions involving global variables can be evaluated, and the expression is evaluated as if it were in a separate function with no local variables.

### D.2.2 Cooperative Multitasking

Cooperative multitasking is a convenient way to perform several different tasks at the same time. An example would be to step a machine through a sequence of steps and at the same time independently carry on a dialog with the operator via a human interface. Cooperative multitasking differs from a different approach called preemptive multitasking. Dynamic C supports both types of multitasking. In cooperative multitasking each separate task voluntarily surrenders its compute time when it does not need to perform any more activity immediately. In preemptive multitasking control is forcibly removed from the task via an interrupt.

Dynamic C has language extensions to support multitasking. The major C constructs are called *costatements, cofunctions,* and *slicing*. These are described more completely in the ***Dynamic C User's Manual***. The example below, sample program **`DEMOBRD3.C`**, uses costatements. A costatement is a way to perform a sequence of operations that involve pauses or waits for some external event to take place. A complete description of costatements is in the ***Dynamic C User's Manual***. The **`DEMOBRD3.C`** sample program has two independent tasks. The first task flashes LED2 once a second. The second task uses button SW1 on the Demonstration Board to toggle the logical value of a virtual switch, **`vswitch`**, and flash LED1 each time the button is pressed. This task also debounces button SW1.

Note that the Demonstration Board has to be connected to the Intellicom board as described in Section D.1 to be able to run **`DEMOBRD3.C`**.

```
main() {
    int vswitch;        // state of virtual switch controlled by button S1

    WrPortI(PDDDR, &PDDDRShadow, 0x03); // set port D bits 0-1 as outputs
    WrPortI(PDDCR, &PDDCRShadow, 0x00); // set port D to not open drain mode
    vswitch = 0;                        // initialize virtual switch as off

(1) while (1) {                         // endless loop

        // First task will flash LED4 for 200 ms once per second.

(2)     costate {
            BitWrPortI(PDDR, &PDDRShadow, 0xFF, 1);    // turn LED on
(3)         waitfor(DelayMs(200));                     // wait 200 ms
            BitWrPortI(PDDR, &PDDRShadow, 0x00, 1);    // turn LED off
            waitfor(DelayMs(800));                     // wait 800 ms
(4)     }

        // Second task - debounce SW1 and toggle vswitch

        costate {
(5)         if (!BitRdPortI(PDDR, 2)) abort; // if button not down skip out
            waitfor(DelayMs(50));            // wait 50 ms
            if(!BitRdPortI(PDDR, 2))  abort; // if button not still down exit

            vswitch = !vswitch;       // toggle since button was down 50 ms

            while (1) {
                waitfor(!BitRdPortI(PDDR, 2));  // wait for button to go up
                waitfor(DelayMs(200));          // wait additional 200 ms
                if (!BitRdPortI(PDDR, 2))
                    break;    // if button still up break out of while loop
            }
        }                     // end of costate

        // make LED1 agree with vswitch

(6)     BitWrPortI(PDDR, &PDDRShadow, vswitch, 0);

(7) }   // end of while loop
}       // end of main
```

The numbers in the left margin are reference indicators, and are not a part of the code. Load and run the program. Note that LED2 flashes once per second. Push button SW1 several times and note how LED1 is toggled.

The flashing of LED2 is performed by the costatement starting at the line marked (2). Costatements need to be executed regularly, often at least every 25 ms. To accomplish this, the costatements are enclosed in a **`while`** loop. The term **`while`** loop is used as a handy way to describe a style of real-time programming in which most operations are done in one loop. The while loop starts at (1) and ends at (7).

The statement at (3) waits for a time delay, in this case 200 ms. The costatement is being executed on each pass through the big loop. When a **waitfor** condition is encountered the first time, the current value of **MS_TIMER** is saved and then on each subsequent pass the saved value is compared to the current value. If a **waitfor** condition is not encountered, then a jump is made to the end of the costatement (4), and on the next pass of the loop, when the execution thread reaches the beginning of the costatement, execution passes directly to the **waitfor** statement. Once 200 ms has passed, the statement after the waitfor is executed. The costatement has the property that it can wait for long periods of time, but not use a lot of execution time. Each costatement is a little program with its own statement pointer that advances in response to conditions. On each pass through the big loop, as little as one statement in the costatement is executed, starting at the current position of the costatement's statement pointer. Consult the ***Dynamic C User's Manual*** for more details.

The second costatement in the program debounces the switch and maintains the variable **vswitch**. Debouncing is performed by making sure that the switch is either on or off for a long enough period of time to ensure that high-frequency electrical hash generated when the switch contacts open or close does not affect the state of the switch. The **abort** statement is illustrated at (5). If executed, the internal statement pointer is set back to the first statement within the costatement, and a jump to the closing brace of the costatement is made.

At (6) a use for a shadow register is illustrated. A shadow register is used to keep track of the contents of an I/O port that is write only - it can't be read back. If every time a write is made to the port the same bits are set in the shadow register, then the shadow register has the same data as the port register. In this case a test is made to see the state of the LED and make it agree with the state of vswitch. This test is not strictly necessary, the output register could be set every time to agree with vswitch, but it is placed here to illustrate the concept of a shadow register.

To illustrate the use of snooping, use the watch window to observe **vswitch** while the program is running. Add the variable **vswitch** to the list of watch expressions. Then toggle **vswitch** and the LED. Then type **<ctrl-U>** to observe **vswitch** again.

### D.2.3 Advantages of Cooperative Multitasking

Cooperative multitasking, as implemented with language extensions, has the advantage of being intuitive. Unlike preemptive multitasking, variables can be shared between different tasks without having to take elaborate precautions. Sharing variables between tasks is the greatest cause of bugs in programs that use preemptive multitasking. It might seem that the biggest problem would be response time because of the big loop time becoming long as the program grows. Our solution for that is a device caused slicing that is further described in the ***Dynamic C User's Manual***.

# INDEX


## A

AC adapter ..... 3, 9

## B

backup battery board ..... 70
  replacing ..... 70
battery backup circuit ..... 68
battery connections ..... 68
battery life ..... 68
bezel ..... 61
  disassembling Intellicom .. 61
board initialization
  function calls ..... 37
    brdInit ..... 37

## C

CE compliance ..... 4
  design guidelines ..... 5
chip select circuit ..... 71
clock doubler ..... 30
conduit ..... 65
conformal coating ..... 56
connections
  Ethernet cable ..... 46
  power supply ..... 8, 9
  programming cable ..... 12

## D

Demonstration Board ..... 3
  hookup instructions ..... 74
  jumper configurations ..... 74
demonstration program ..... 10
digital I/O
  function calls ..... 37
    digIn ..... 37
    digOut ..... 37
  SMODE0 ..... 26
  SMODE1 ..... 26
digital inputs ..... 19
  pullup/pulldown configuration ..... 19
digital outputs ..... 19
  sinking ..... 19
dimensions
  bezel ..... 62
  front panel ..... 62
  Intellicom board ..... 54
  outer casing ..... 63
display
  vacuum fluorescent display option ..... 2, 29
display controls
  function calls ..... 41
    dispClear ..... 42
    dispContrast ..... 41
    dispCursor ..... 41
    dispGoto ..... 41
    dispOnoff ..... 42
    dispPrintf ..... 42
    dispPutc ..... 42
Dynamic C ..... 3, 31
  break point ..... 76
  COM port ..... 14
  debugging features ..... 31
  Dynamic C modules ..... 33
  editing the program ..... 77
  features ..... 77
  handling different memories in BIOS ..... 28
  libraries ..... 36
  memory
    BIOS ..... 28
  multitasking ..... 78, 80
  Rabbit Embedded Security Pack ..... 3
  single-stepping ..... 76
  standard features ..... 32
    debugging ..... 32
  starting ..... 14
  telephone-based technical support ..... 3
  upgrades and patches ..... 33
  watch expression ..... 76
  watching variables dynamically ..... 77

## E

EMI
  spectrum spreader feature . 30
Ethernet cables ..... 45, 65
Ethernet connections ..... 45, 46
  Ethernet cable ..... 65
  steps ..... 45

## F

features ..... 1
flash memory
  liefetime write cycles ..... 31
front panel bezel dimensions 62

## H

headers
  Demonstration Board
    H1 ..... 74
    H2 ..... 74
  JP1 ..... 25

## I

I/O pinout ..... 18
Intellicom
  features
    software demonstration . 11
  introduction ..... 1
introduction ..... 1
IP addresses ..... 50
  how to set ..... 49
  how to set PC IP address .. 50

## J

jumper configurations ..... 57, 58
  Demonstration Board ..... 74
  JP1 (RS-485 bias and termination resistors) ..... 25
  JP1 (SRAM size) ..... 58
  JP2 (flash memory bank select) ..... 28
  JP2 (flash memory size) .... 58
  JP3 (flash memory size) .... 58

jumper configurations (cont'd)
JP4 (digital input pull-up/pull-down resistors) .....58
JP5 (flash memory bank select) .....58
JP6 (RS-485 bias and termination resistors) .....58
JP7 (RS-232/RS-485 select) .....58
jumper locations .....57

## K

keypad
disassembling from bezel ..61
keypad controls
function calls .....39
keyConfig .....39
keyGet .....40
keyInit .....40
keypadDef .....40
keyProcess .....39
keypad insert .....60
changing .....60
keypad matrix .....39

## M

memory .....28
BIOS .....28
flash EPROM configuration for different sizes .....28
SRAM configuration for different sizes .....28
models .....2
OP6600 .....2
OP6700 .....2

## O

OP6600 .....2
OP6700 .....2
outer casing
removal .....8

## P

pinout
Ethernet port .....18
I/O .....18
RJ-12 connector .....18
RJ-45 connector .....18
serial communication .....18
plastic casing .....62, 63, 65
assembly .....64, 65
attaching conduit .....65
Ethernet cable .....65
mounting .....65
mounting brackets .....64
splash-resistant mounting ..64
power management .....67
power supplies .....67
Backup battery board .....68
battery backup .....68
battery backup circuit .....68
battery life .....68
chip select circuit .....71
switching voltage regulator 67
VRAM switch .....69
power-up
demonstration program .....10
programming
flash vs. RAM .....31
programming cable .....3, 12
programming port .....26
programming cable
PROG connector .....27

## R

reset .....9, 12
reset generator .....70
RS-232 .....23
RS-485 .....23
termination and bias resistors 25
RS-485 network .....24
running TCP/IP sample programs .....47

## S

sample programs .....34
COFTERMA.C .....34
DEMOBRD1.C .....34, 75, 76
DEMOBRD2.C .....34, 77
DEMOBRD3.C .....34, 78
how to run .....73
how to set IP address .....49
HTTPDEMO.C .....34
ICOM232.C .....34
ICOM485.C .....34
ICOM5WIRE.C .....34
ICOMDEMO.C .....11, 34
ICOMIO.C .....34
KEYLCD.C .....34
MBOXDEMO.C .....35
MUSIC.C .....35
MUSIC2.C .....35
PINGME.C .....51
PONG.C .....15
power-up demonstration program .....10
REMOTE1.C .....35
running TCP/IP sample programs .....47
RXSAMPLE.C .....51
SAMPLES\ICOM\WINDOWS directory ...35
SMTPDEMO.C .....35
SPEAKER.C .....35
SSI3.C .....51
STATIC.C .....51
TCP/IP .....47
TCP_RESPOND.C .....35
TCP_SEND.C .....35
using PCRESPOND .....35
using PCSEND .....35
serial communication .....20, 38
function calls
serB485Rx .....38
serB485Tx .....38
serMode .....38
programming port .....26
RS-232 description .....23
RS-232/RS-485 options .....20, 21, 22
RS-485 description .....23
RS-485 network .....23, 24
common power supply ..23
RS-485 termination and bias resistors .....25
serial communication pinout .18
software .....36
libraries
ICOM.LIB .....36
PACKET.LIB .....38
RS232.LIB .....38
TCPIP.LIB .....36
speaker .....29
output characteristics .....29
speaker controls
function calls .....44
spkrOut .....44
specifications .....53
electrical .....55
front panel bezel dimensions .....62
mechanical dimensions
Intellicom board .....54
outer casing dimensions ....63
temperature .....55

spectrum spreader ..... 30
splash-resistant mounting ..... 64
subsystems ..... 18

## T

TCP/IP connections .. 45, 46, 48
10Base-T ..... 47
10Base-T Ethernet card .... 45
additional resources ..... 52
Ethernet cable ..... 65
Ethernet cables ..... 47
Ethernet hub ..... 45
IP addresses ..... 47
steps ..... 45
template
keypad insert ..... 60
Tool Kit ..... 3
AC adapter ..... 3
Demonstration Board ..... 3
Dynamic C software ..... 3
programming cable ..... 3
software ..... 3
User's Manual ..... 3
wire assembly ..... 3

## U

USB/serial port converter ..... 12
Dynamic C settings ..... 14

## V

vacuum fluorescent display 2, 29
changing PCB configuration ..... 29

# SCHEMATICS

## 090-0095 Intellicom Schematic

www.rabbit.com/documentation/schemat/090-0095.pdf

## 090-0042 Demonstration Board Schematic

www.rabbit.com/documentation/schemat/090-0042.pdf

## 090-0128 Programming Cable Schematic

www.rabbit.com/documentation/schemat/090-0128.pdf

You may use the URL information provided above to access the latest schematics directly.

Стандарт Электрон Связь

Мы молодая и активно развивающаяся компания в области поставок электронных компонентов. Мы поставляем электронные компоненты отечественного и импортного производства напрямую от производителей и с крупнейших складов мира.

Благодаря сотрудничеству с мировыми поставщиками мы осуществляем комплексные и плановые поставки широчайшего спектра электронных компонентов.

Собственная эффективная логистика и склад в обеспечивает надежную поставку продукции в точно указанные сроки по всей России.

Мы осуществляем техническую поддержку нашим клиентам и предпродажную проверку качества продукции. На все поставляемые продукты мы предоставляем гарантию .

Осуществляем поставки продукции под контролем ВП МО РФ на предприятия военно-промышленного комплекса России , а также работаем в рамках 275 ФЗ с открытием отдельных счетов в уполномоченном банке. Система менеджмента качества компании соответствует требованиям ГОСТ ISO 9001.

Минимальные сроки поставки, гибкие цены, неограниченный ассортимент и индивидуальный подход к клиентам являются основой для выстраивания долгосрочного и эффективного сотрудничества с предприятиями радиоэлектронной промышленности, предприятиями ВПК и научно-исследовательскими институтами России.

С нами вы становитесь еще успешнее!

**Наши контакты:**

**Телефон:** +7 812 627 14 35

**Электронная почта:** sales@st-electron.ru

**Адрес:** 198099, Санкт-Петербург, Промышленная ул, дом № 19, литера Н, помещение 100-Н Офис 331