Intellicom

Models OP6600 and OP6700

User's Manual

000915 - A

Intellicom User's Manual

Part Number 019-0078 • 000915 - A • Printed in U.S.A.

CopyrightPke c[(P)4(hig)-4.anges [(P)4(n)2.PmPmtg notice[(72.(in U.S.A.)]TJET5672.045 656.865 4

Table of Contents

About This Manual

This manual provides instructions for installing, testing, configuring, and interconnecting the Intellicom operator interface.

Assumptions

Assumptions are made regarding the user's knowledge and experience in the following areas:

- Ability to design and engineer the target system that an Intellicom will ineract with.
- Understanding of the basics of operating a software program and editing files under Windows on a PC.
- Knowledge of basic assembly language and architecture for controllers.

For a full treatment of C, refer to the following texts:

The C Programming Language by Kernighan and Ritchie (published by Prentice-Hall).

and/or

C: A Reference Manual by Harbison and Steel (published by Prentice-Hall).

- Knowledge of basic assembly language and Rabbit microprocessor architecture.

For more information on the Rabbit 2000 microprocessor, refer to the ***Rabbit 2000 Microprocessor User's Guide***.

Acronyms

Table 1 lists and defines the acronyms that may be used in this manual.

Table 1. Acronyms

Acronym	Meaning
EPROM	Erasable Programmable Read-Only Memory
EEPROM	Electronically Erasable Programmable Read-Only Memory
FD	Factory Default
NMI	Nonmaskable interrupt
PIO	Parallel Input/Output (Individually programmable input/output)
PRT	Programmable Reload Timer
RAM	Random Access Memory
RTC	Real-Time Clock
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver Transmitter

Conventions

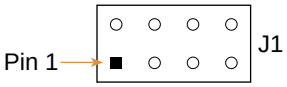
Table 2 lists and defines the typographic conventions that may be used in this manual.

Table 2. Typographic Conventions

Example	Description
`while`	Bold Courier font indicates a program, a fragment of a program, or a Dynamic C keyword or phrase.
`// IN-01…`	Program comments are in normal Courier font.
Italics	Courier italics indicate that something should be typed instead of the italicized words (e.g., type a file name where *`filename`* is shown).
Edit	Bold sans serif font indicates a menu or menu selection.
…	An ellipsis indicates that (1) irrelevant program text is omitted for brevity, or that (2) the preceding program text may be repeated indefinitely.
[ ]	Square brackets in a C function's definition or program segment indicate that the enclosed directive is optional.
< >	Angle brackets are used to enclose classes of terms.
a | b | c	A vertical bar indicates that a choice should be made from among the items listed.

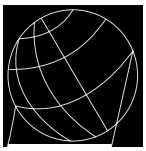
Pin Number 1

A black square indicates pin 1 of all headers.



Measurements

All diagram and graphic measurements are in inches followed by millimeters enclosed in parenthesis.



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
- 10BaseT 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
- Water-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.

1.2 Development and Evaluation Tools

1.2.1 Development Kit

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

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

- ***Intellicom User's Manual*** with schematics.
- 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.
-

The disadvantage of using flash is that interrupts must be disabled for approximately 5 ms whenever a break point is set in the program. This can crash fast interrupt routines that are running while you stop at a break point or single-step the program. Flash or RAM is selected on the **Options** > **Compiler** menu.

Dynamic C Premier provides a number of debugging features. You can single-step your program, either in C, statement by statement, or in assembly language, instruction by instruction. You can set break points, where the program will stop, on any statement. You can evaluate watch expressions. A watch expression is any C expression that can be evaluated in the context of the program. If the program is at a break point, a watch expression can view any expression using local or external variables. If the program is running and a call to the debugger is included in the user's code (`runwatch();`), it is possible to evaluate watch expressions using global variables only while the target program continues to run, slowed down only by the need to refresh a display in response to a **<Ctrl-U>** command.

2. GETTING STARTED


SRAM
Flash
EPROM
Flash
EPROM
Z-COM
PWR

3. Connect Power Supply to the Intellicom Board.

Connect the positive lead (indicated with red heat-shrink tubing on the AC adapter included with the Development 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.

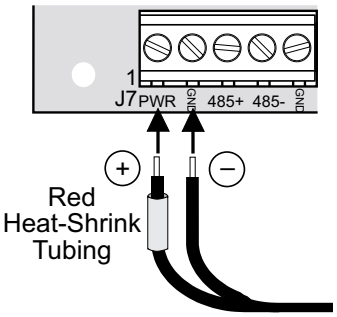


Figure 3. Power Supply Connections—Detailed View

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

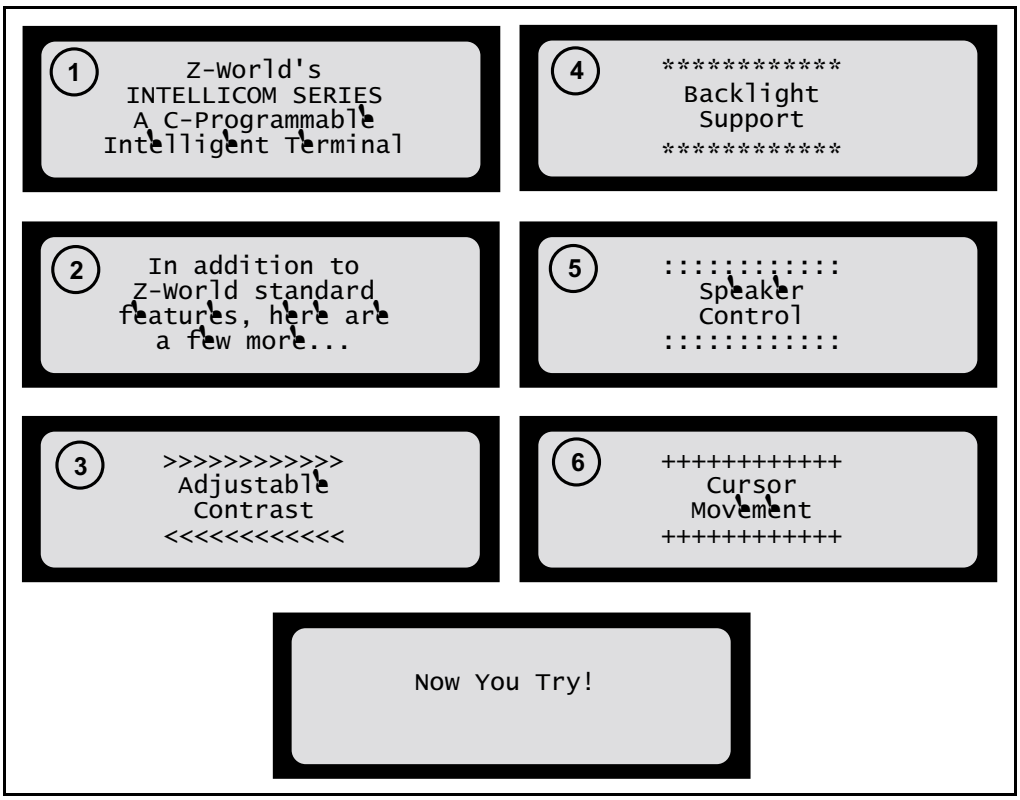
4. Apply power.

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

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

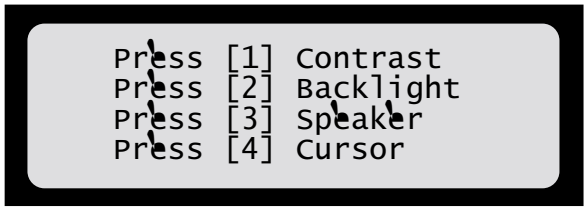
2.2 Demonstration Program on Power-Up

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.

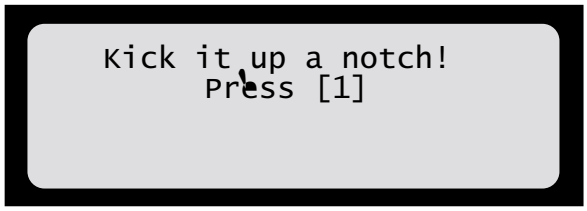


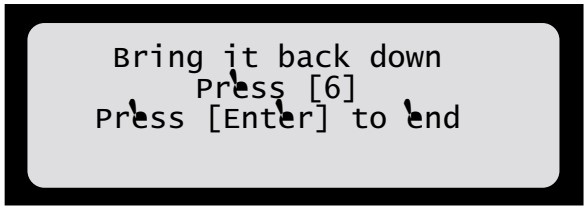
The contrast, backlight, speaker volume, and cursor positions will change automatically through the demonstration. Then there is an opportunity for you to vary these settings by responding to prompts on the LCD.

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

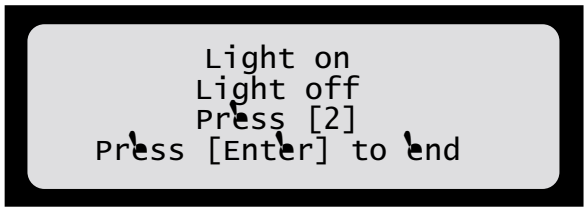


2. Press [**1**] to select the contrast adjustment demonstration.
3. Press [**1**] to increase contrast, press [**6**] to decrease contrast, or press [**Enter**] to get to choose another feature.

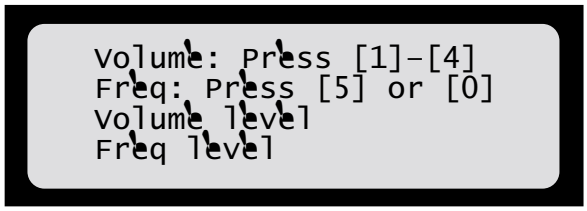




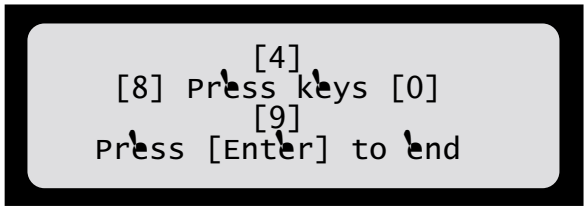
4. Press [**2**] to select the backlight demonstration. Press [**2**] to toggle backlight on or off, or press [**Enter**] to get to choose another feature.



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 Premier **SAMPLES** directory as **ICOMDEMO.C**.

2.3 Programming Cable Connections

1. Connect the Programming Cable to the Intellicom Board.

Connect the 10-pin connector of the programming cable to header J4 on the Intellicom Board as shown in Figure 4. Connect the other end of the programming cable to a COM port on your PC. Note that COM1 on the PC is the default COM port used by Dynamic C.

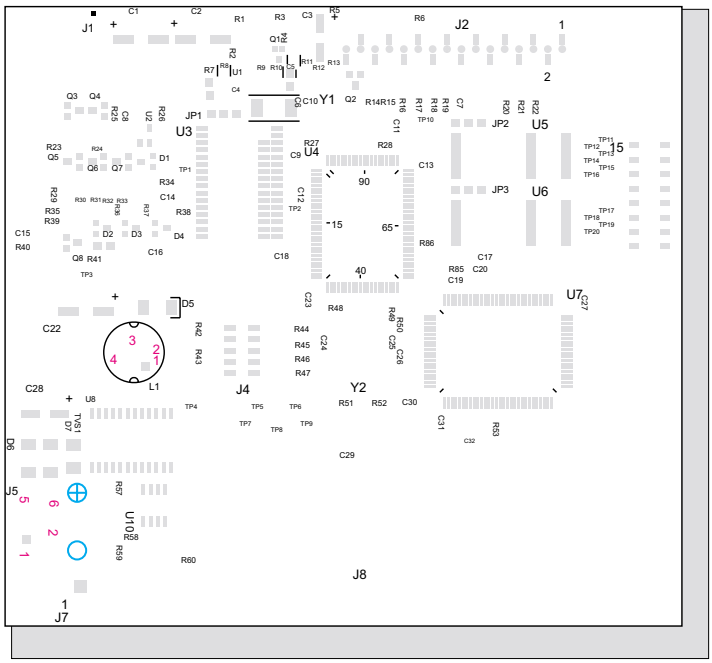


Figure 4. Programming Cable Connections

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.

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

2.4 Installing Dynamic C

If you have not yet installed Dynamic C, you may do so by inserting the Dynamic C Premier CD 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 **setup.exe** file on your CD drive. Click **OK** to begin the installation once you have selected the **setup.exe** file.

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

2.5 Starting Dynamic C

Once the Intellicom board is connected as described above, start Dynamic C by double-clicking on the Dynamic C icon or by double-clicking on the **.exe** file associated with **DcRab** in the Dynamic C directory.

Dynamic C assumes, by default, that you are using serial port COM1 on your PC. If you ***are*** using COM1, then Dynamic C should detect the Intellicom board and go through a sequence of steps to cold-boot the Intellicom board and to compile the BIOS. If the error message "Rabbit Processor Not Detected" appears, you have probably connected to a different PC serial port such as COM2, COM3, or COM4. You can change the serial port used by Dynamic C with the **OPTIONS** menu, then try to get Dynamic C to recognize the Intellicom board by selecting **Recompile BIOS** on the **Compile** menu. Try the different COM ports in the **OPTIONS** menu until you find the one you are connected to. If you still can't get Dynamic C to recognize the target on any port, then the hookup may be wrong or the COM port is not working on your PC.

If you receive the "BIOS successfully compiled …" message after pressing **<Ctrl-Y>** or starting Dynamic C, and this message is followed by "Target not responding," it is possible that your PC cannot handle the 115,200 bps baud rate. Try changing the baud rate to 57,600 bps as follows.

1. Open the BIOS source code file named **RABBITBIOS.C**, which can be found in the **BIOS** directory.
2. Change the line

   ```
   #define USE115KBAUD 1   // set to 0 to use 57600 baud
   ```

   to read as follows.

   ```
   #define USE115KBAUD 0   // set to 0 to use 57600 baud
   ```

3. Locate the **Serial options** dialog in the Dynamic C **Options** menu. Change the baud rate to 57,600 bps, then press **<Ctrl-Y>**.

When you receive the "BIOS successfully compiled …" message and do not receive a "Target not responding" message, the target is now ready to compile a program.

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 5.8, "Run the PINGME.C Demo," tests the TCP/IP portion of the board.

2.7 Where Do I Go From Here?

If there are any problems at this point, call Z-World Technical Support at (530)757-3737.

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.

- Switching Between Program Mode and Run Mode
- Intellicom Subsystems
- Serial Communication
- Memory
- Speaker

3.1 Switching Between Program Mode and Run Mode

The Intellicom is automatically in Program Mode when the programming cable is attached, and is automatically in Run Mode when no programming cable is attached. See Figure 5.

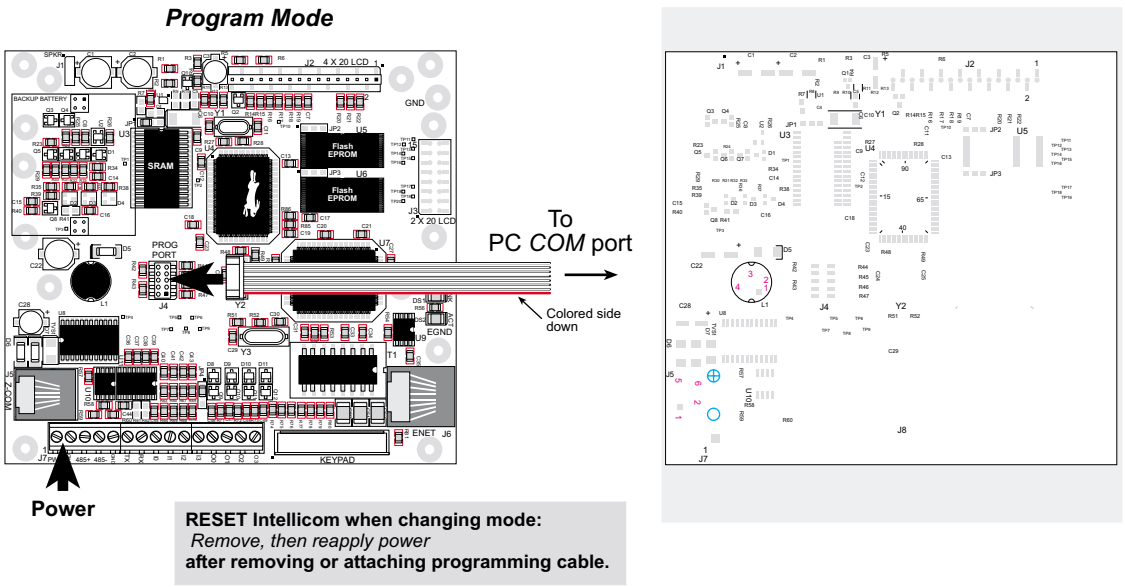


Figure 5. Intellicom Program Mode and Run Mode Set-Up

3.1.1 Detailed Instructions: Changing from Program Mode to Run Mode

1. Disconnect the programming cable from header J4 of the Intellicom board.
2. Reset the Intellicom by unplugging the AC adapter, then plugging it back in.

The Intellicom is now ready to operate in the Run Mode.

3.1.2 Detailed Instructions: Changing from Run Mode to Program Mode

1. Attach the programming cable to header J4 of the Intellicom board.
2. Reset the Intellicom by unplugging the AC adapter, then plugging it back in.

The Intellicom is now ready to operate in the Program Mode.

3.2 Intellicom Subsystems

Figure 6 shows the Rabbit-based subsystems designed into the Intellicom.

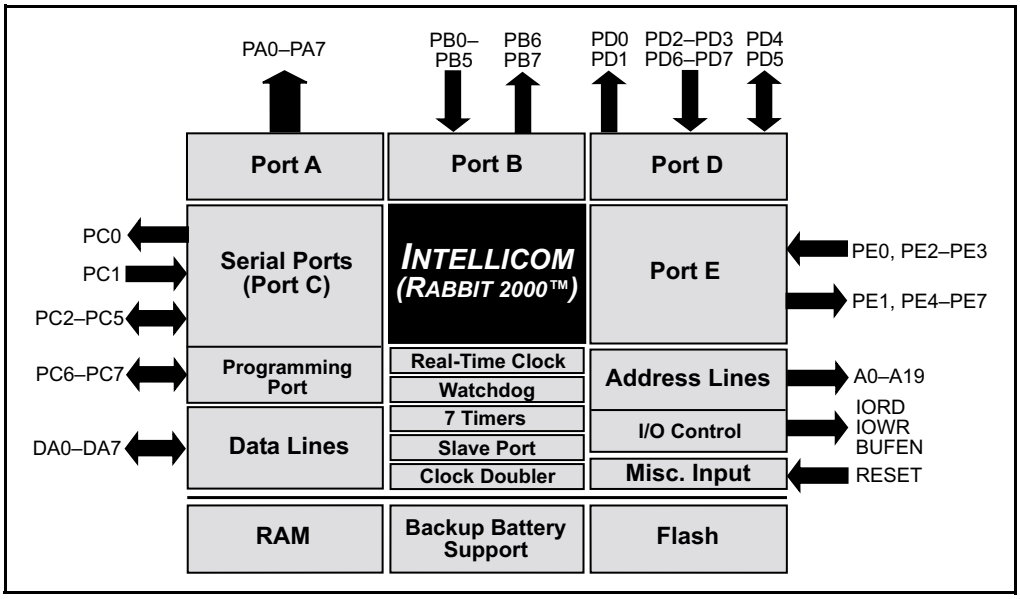


Figure 6. Intellicom Rabbit-Based Subsystems

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 7.

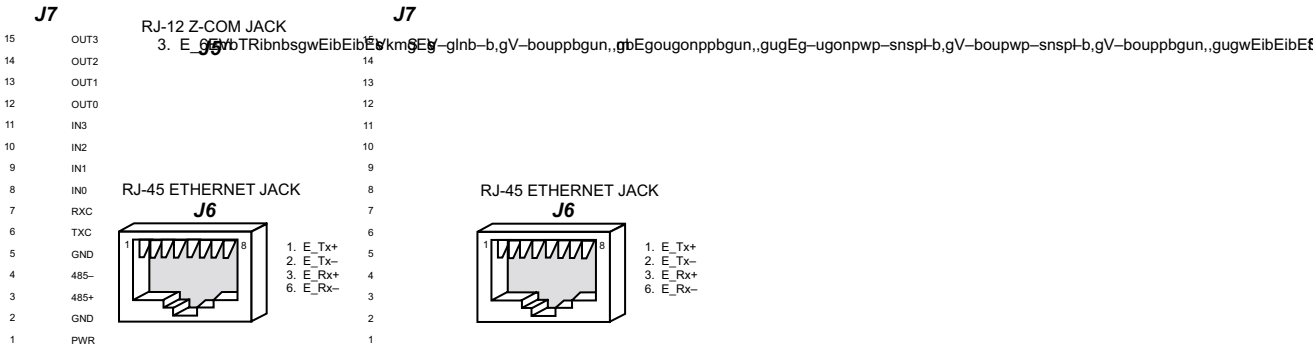


Figure 7. Intellicom I/O Pinout

RJ-45 pinouts are sometimes numbered opposite to the way shown in Figure 7. 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 7) are always absolute, and the RJ-45 connector will work properly with off-the-shelf Ethernet cables.

3.2.1 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 8.

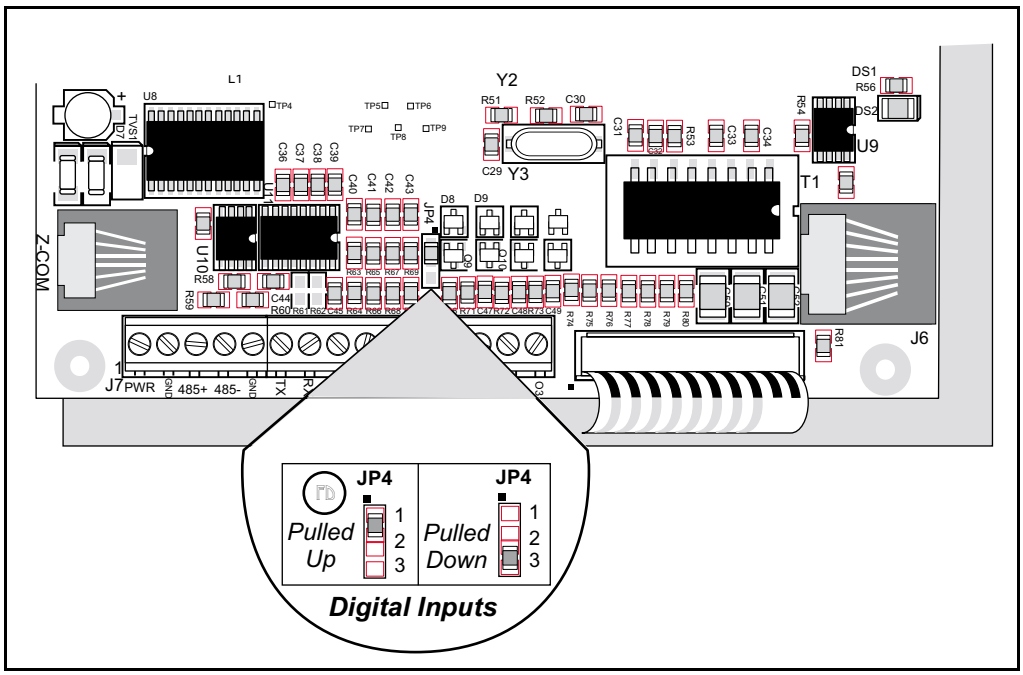


Figure 8. Surface-Mounted Jumper Configurations for Selecting Pullup/Pulldown on the Digital Inputs

3.2.2 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 VDC.

3.3 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 RS-232 transceiver may be converted to a 5-wire RS-232 channel or 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 9. The RS-485 chip (U10) and the associated bias and termination resistors (R58, R59, and R60) shown in Figure 9 must be removed when configuring the Intellicom for either one 5-wire RS-232 or two 3-wire RS-232.

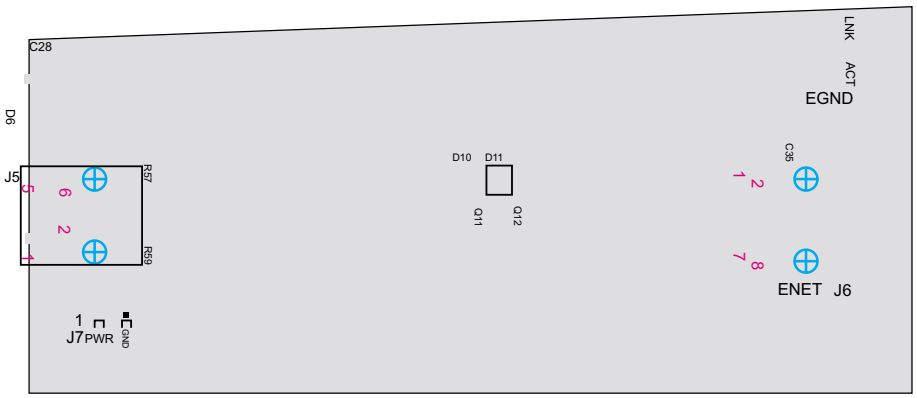


Figure 9. Intellicom RS-232/RS-485 Serial Communication Options

Table 2 provides a summary of 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. Serial Communication Configurations

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

3.3.1 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.

The Rabbit 2000 serial port D ATXB and ARXB signals are normally presented as RS-485, but may be presented as RS-232 TX and RX on pins 3 and 4 of header J7 by adding 0 Ω surface-mounted resistors at R61 and R62 as shown in Figure 9. This leaves the Rabbit 2000 serial port C TXC and RXC signals available to used as a general digital I/O for RTS/CTS handshaking or as a second 3-wire RS-232 channel. The mode is selected with the Dynamic C function **`serMode`**.

The RS-485 chip (U10) and the associated bias and termination resistors (R58, R59, and R60) shown in Figure 9 must be removed when configuring the Intellicom for 5-wire RS-232 or two 3-wire RS-232 channels.

3.3.2 RS-485

The Intellicom has one RS-485 serial channel, which is connected to the Rabbit 2000 serial port D through U10, an RS-485 transceiver. U10 supports the RS-485 serial communication protocol. The chip's slew rate limiters provide for a maximum baud rate of 250,000 bps, which allows for a network of up to 300 m (or 1000 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 10.

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.

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.

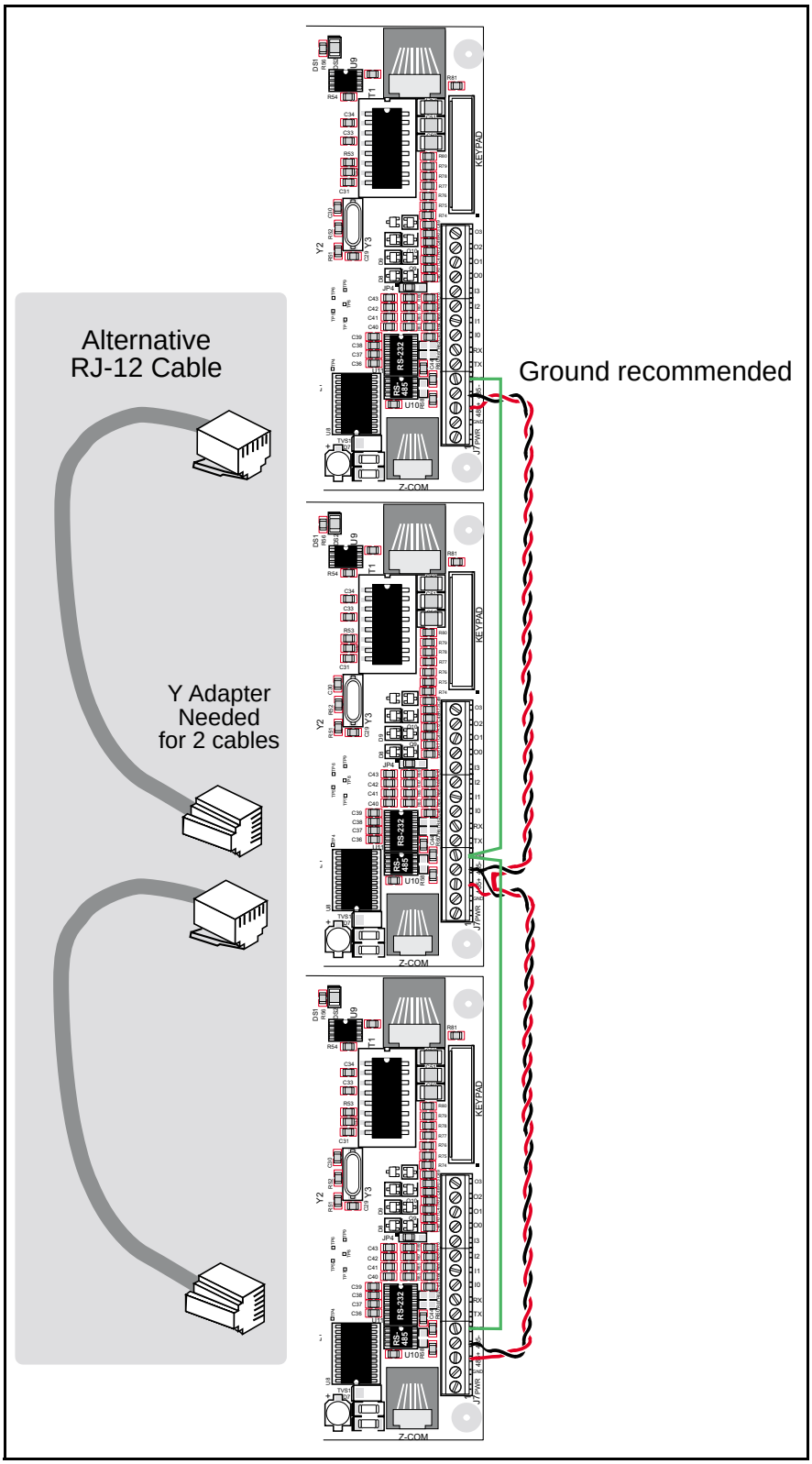


Figure 10. Multidrop Intellicom Network

The Intellicom comes with a 220 Ω termination resistor and 680 Ω bias resistors already installed, as shown in Figure 11.

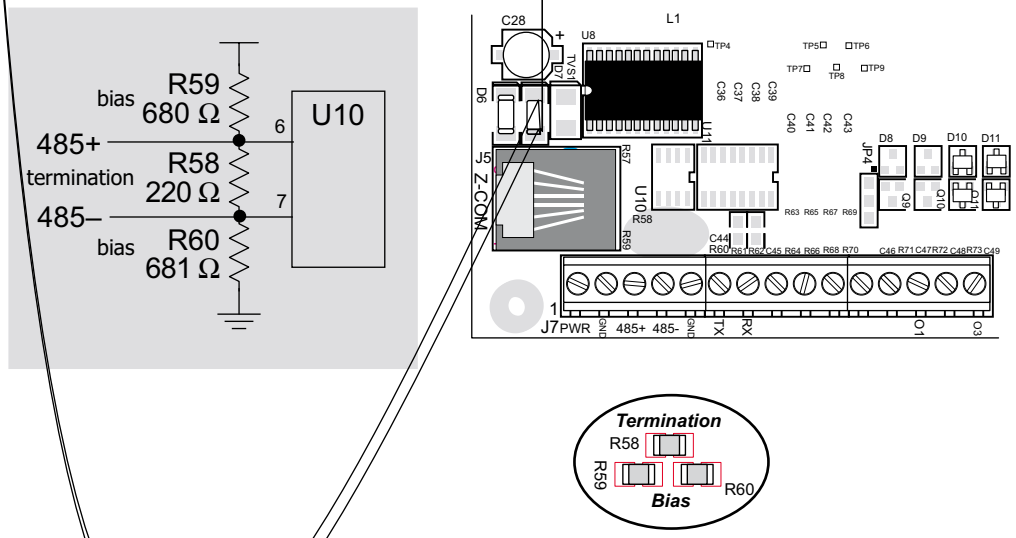


Figure 11. RS-485 Termination and Bias Resistors

The load these bias and termination resistors present to the RS-485 transceiver (U10) limits the number of Intellicom units in a multidrop network to one master and nine slaves, unless the bias and termination resistors are removed. When using more than 10 Intellicom units in a multidrop network, leave the 680 Ω bias resistors in place on the master Intellicom unit, and leave the 220 Ω termination resistors in place on the Intellicom unit at each end of the network.

3.3.3 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 Rabbit 2000 startup-mode pins (SMODE0, SMODE1) are presented to the programming port so that an externally connected device can force the Intellicom to start up in an external bootstrap mode.

> Refer to the ***Rabbit 2000 Microprocessor User's Manual*** for more information related to the bootstrap mode.

The programming port is used to start the Intellicom in a mode where the Intellicom will download a program from the port and then execute the program. The programming port transmits information to and from a PC while a program is being debugged.

The Intellicom can be reset from the programming port.

The Rabbit 2000 status pin is also presented to the programming port. The status pin is an output that can be used to send a general digital signal.

The clock line for serial port A is presented to the programming port, which makes fast serial communication possible.

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.

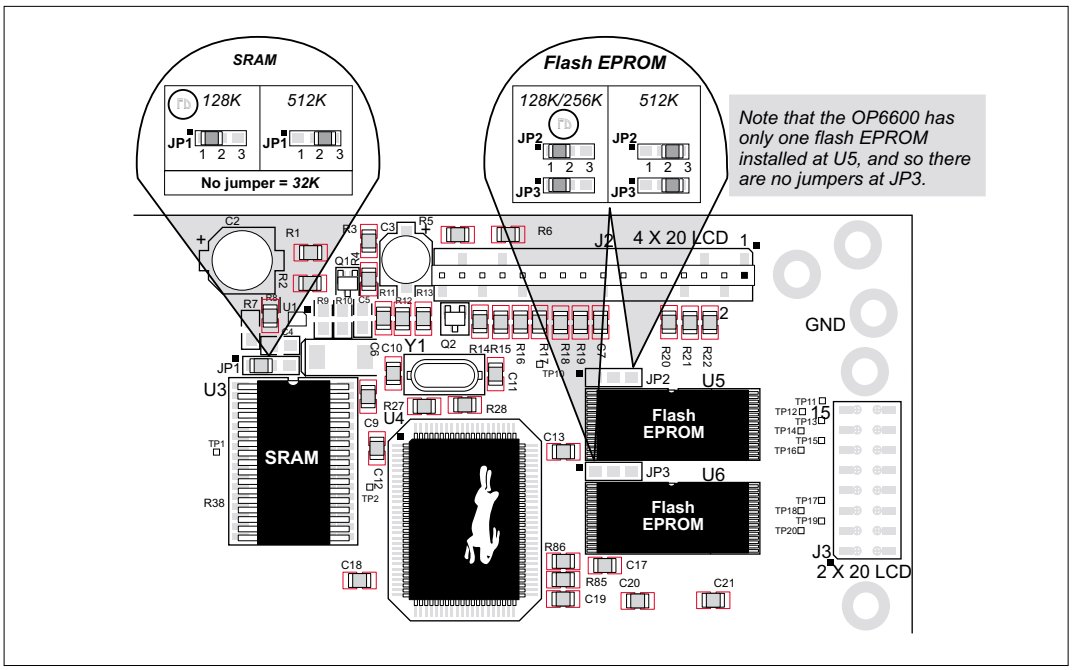


Figure 12. Intellicom Jumper Settings for SRAM and Flash EPROM Size

3.4.2 Flash EPROM

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

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

Z-World 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.

3.4.3 Dynamic C Premier BIOS Source Files

The Dynamic C Premier 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.

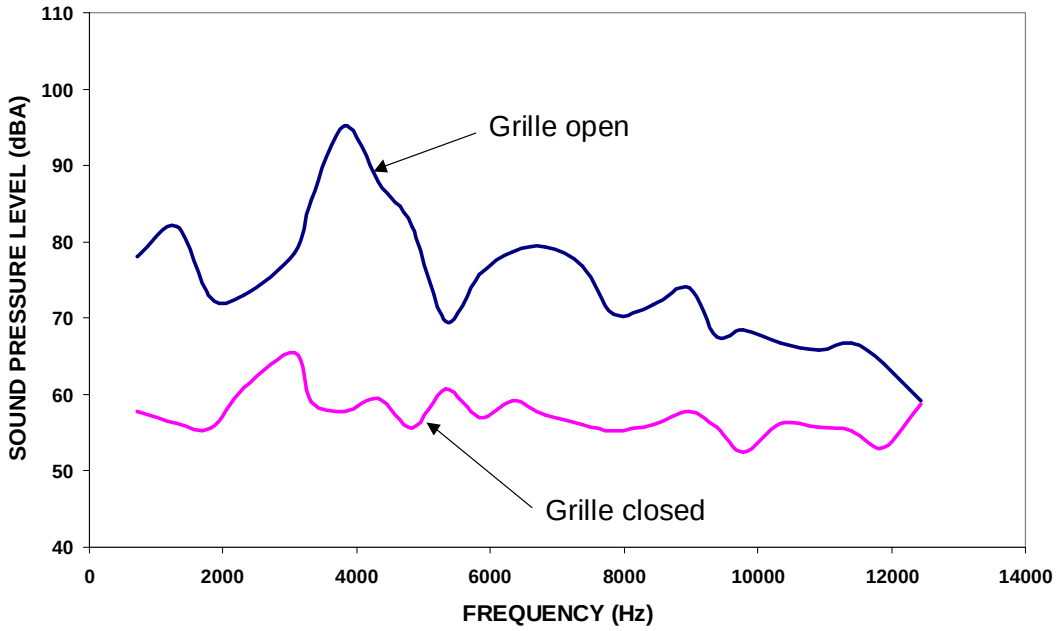


Figure 13. Intellicom Speaker Sound Pressure Level 30 cm (12 inches) from Speaker

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.

Contact your Z-World Sales Representative at (530)7757-3737 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.”

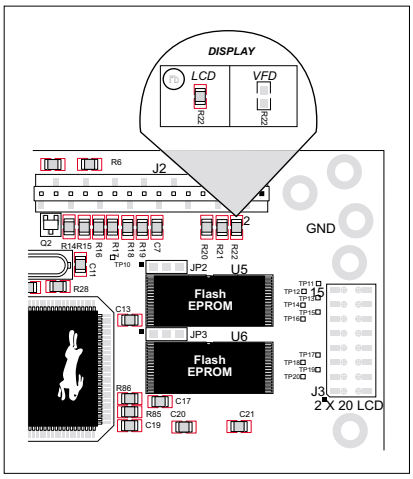


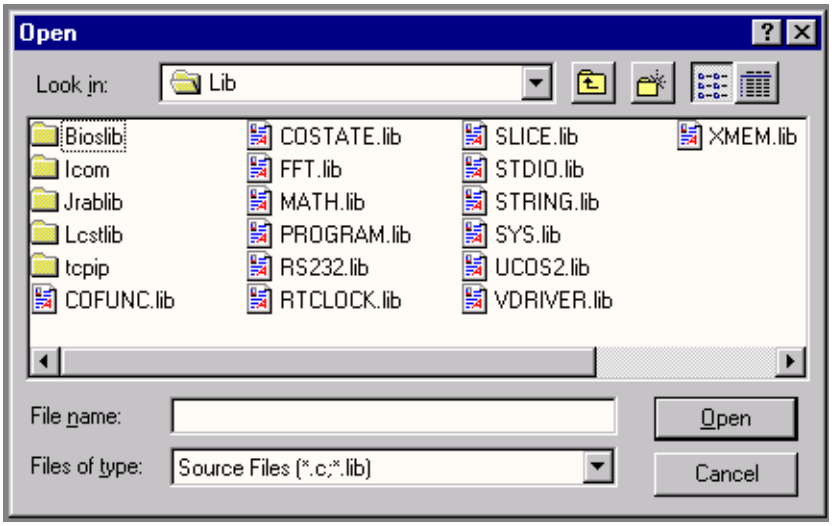
Figure 14. Location of Display Control Resistor, R22

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 Z-World controllers and other controllers based on the Rabbit microprocessor.

4.1 Dynamic C Libraries

With Dynamic C running, click **File** > **Open**, and select **Lib**. The following list of Dynamic C libraries and library directories will be displayed.



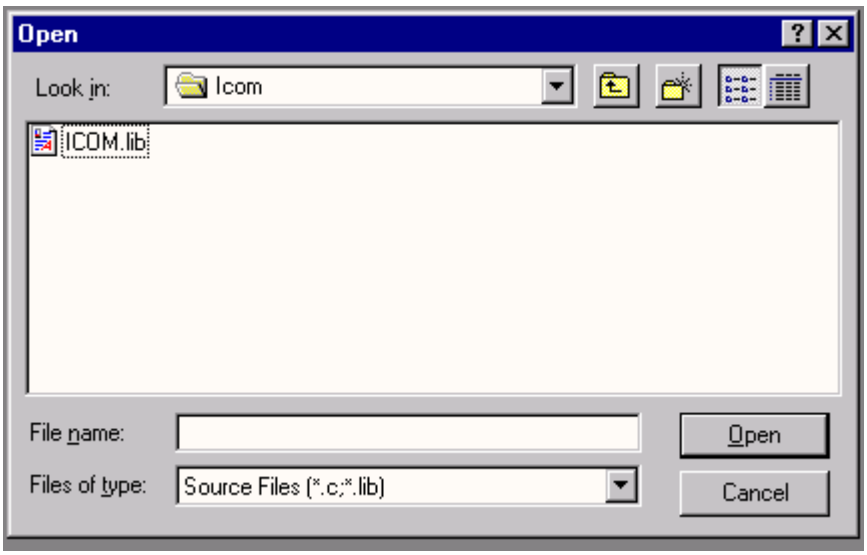
Two library directories are specific to the Intellicom.

- **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 2000 microprocessor are described in the ***Dynamic C Premier User's Manual***.

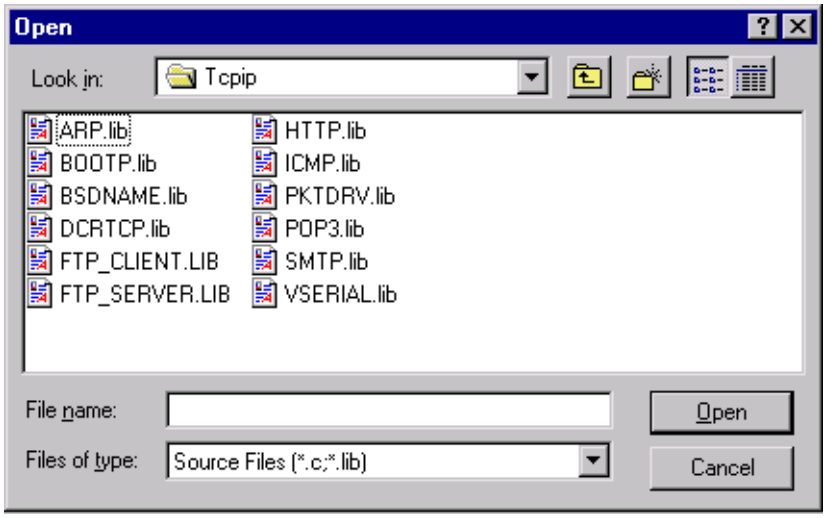
4.1.1 Library Directories

The **ICOM** directory contains libraries required to operate the Intellicom unit.



- **ICOM.LIB**—This library supports OP6600 and OP6700 Intellicom boards only. The functions in this library are described in this chapter.

The **TCPIP** directory contains libraries specific to using TCP/IP functions on the Intellicom board.



- **ARP.lib**—address resolution protocol functions.
- **BOOTP.lib**—bootstrap protocol functions.
- **BSDNAME.lib**—BSD-style socket routines.
- **DCRTCP.lib**—TCP/IP functions.
- **FTP_CLIENT.LIB**—FTP client functions.
- **FTP_SERVER.LIB**—FTP server functions.
- **HTTP.lib**—HTTP handler.
- **ICMP.lib**—ICMP handler.
- **PKTDRV.lib**—packet driver functions.
- **POP3.lib**—POP3 functions.
- **SMTP.lib**—SMTP handler.
- **VSERIAL.lib**—virtual Telnet functions.

The functions in these libraries are described in the ***TCP/IP Function Reference*** and the ***TCP/IP High-Level Protocols*** manuals included with the ***Dynamic C Premier User's Manual***.

4.2 Intellicom Function APIs

4.2.1 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`

4.2.2 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`

4.2.3 Serial Communication

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

User interface to set up 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`

4.2.4 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` isHigh-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`

4.2.5 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`

4.2.6 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.

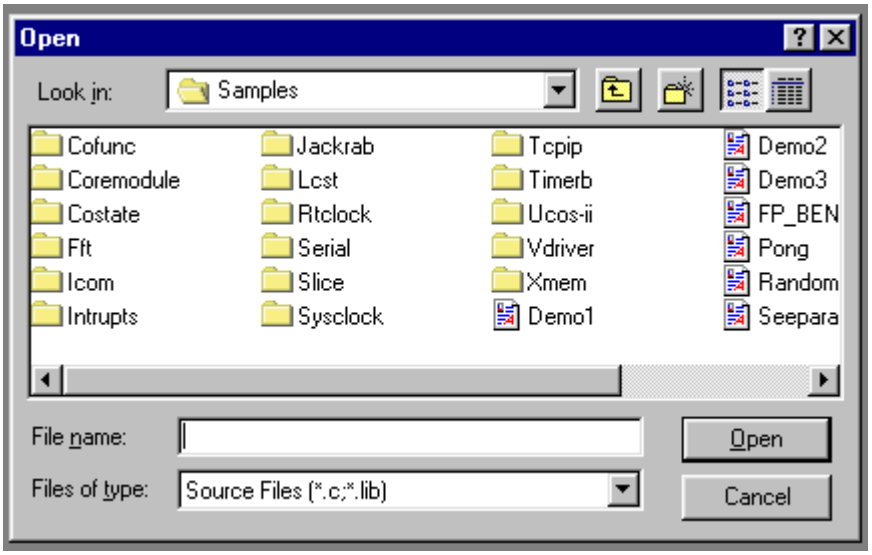
`wAmp`—voltage amplitude (volume) values are 0, 1, 2, and 3: 0 = off, and 3 = loudest volume.

Return Value

None.

4.3 Sample Programs

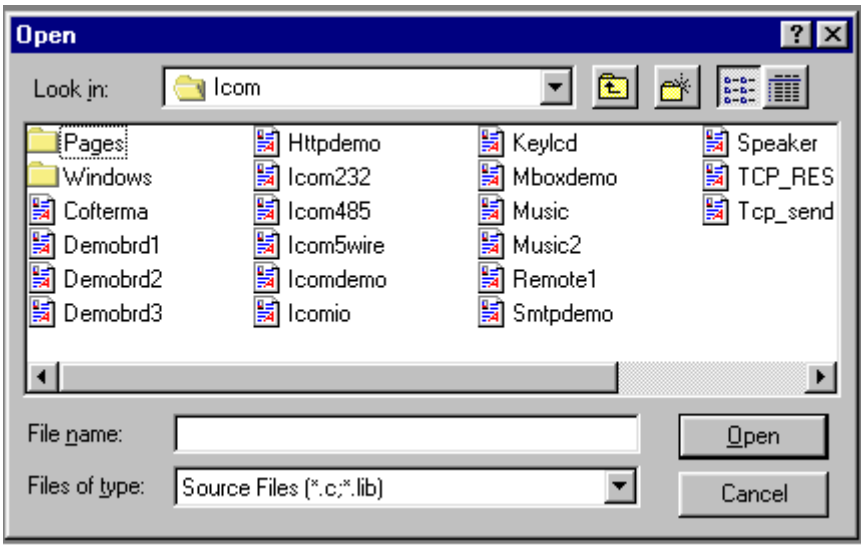
Sample programs are provided in the Dynamic C **Samples** folder, which is shown below.



The various folders contain specific sample programs that illustrate the use of the corresponding Dynamic C libraries. For example, the sample program **PONG.C** demonstrates the output to the **STDIO** window.

The **ICOM** and **TCPIP** folders provide 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.

Let's take a look at the **ICOM** folder.



- **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**.

4.4 Using Dynamic C

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 3.1, “Switching Between Program Mode and Run Mode”) and must be connected to a PC using the programming cable as described in Section 4, “Programming Cable Connections”.

More complete information on Dynamic C is provided in the ***Dynamic C Premier User’s Manual***. TCP/IP specific functions are described in the ***TCP/IP Application Frameworks Manual*** and the ***TCP/IP Function Reference Manual***. Information on using the TCP/IP features and sample programs is provided in Chapter 5., “Using the TCP/IP Features.”

5. USING THE TCP/IP FEATURES

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 10BaseT 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 Z-World in a TCP/IP tool kit. More information is available at www.zworld.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 board to a PC with at least a 10BaseT Ethernet card.

If you have Ethernet access, use a straight Ethernet cable to establish an Ethernet connection to the Intellicom board from an Ethernet hub. These connections are shown in Figure 15.

The PC running Dynamic C through the serial port on the Intellicom board does not need to be the same as the PC with the Ethernet card.

3. Apply Power

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

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.

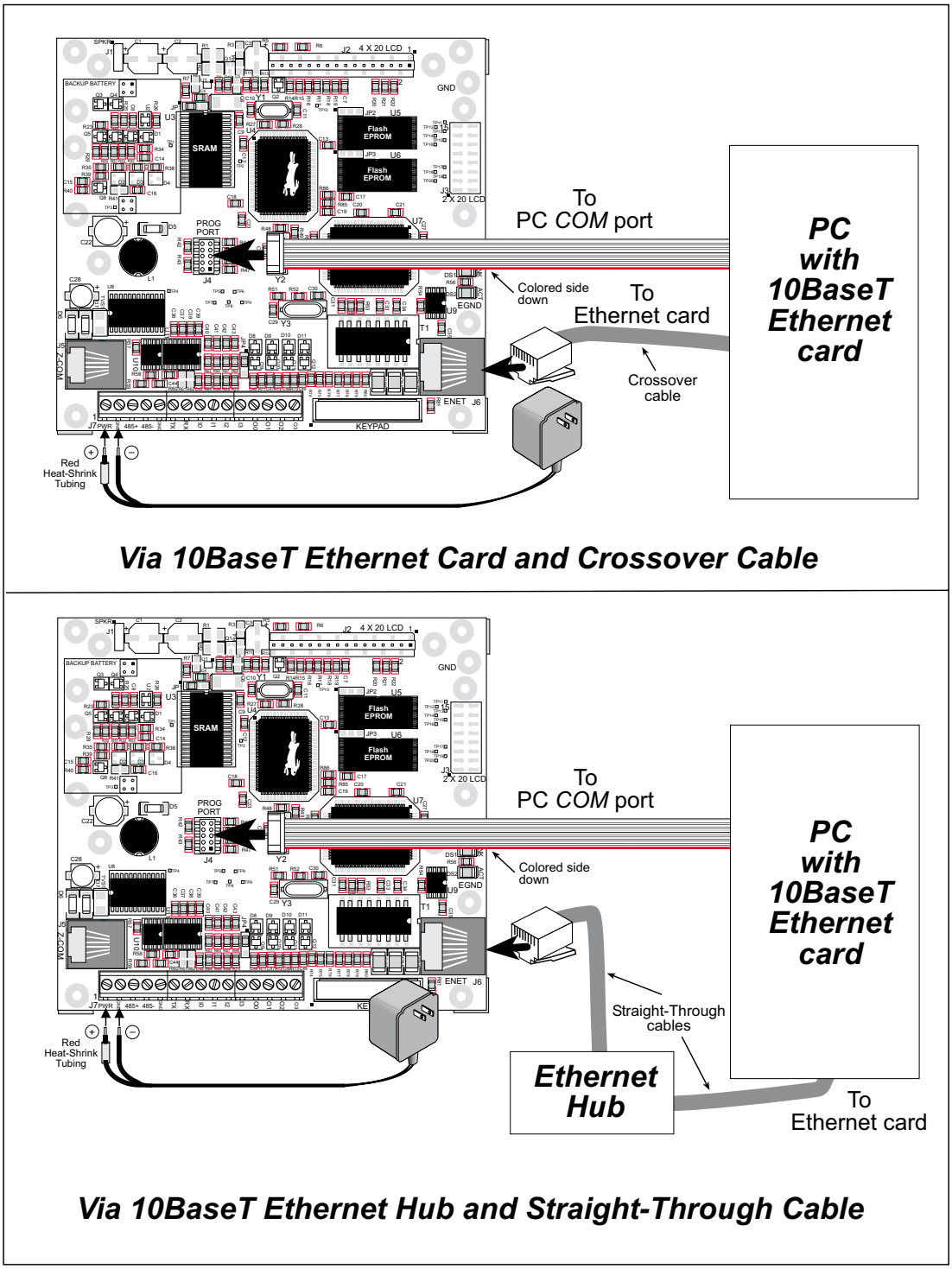


Figure 15. Ethernet Connections

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.

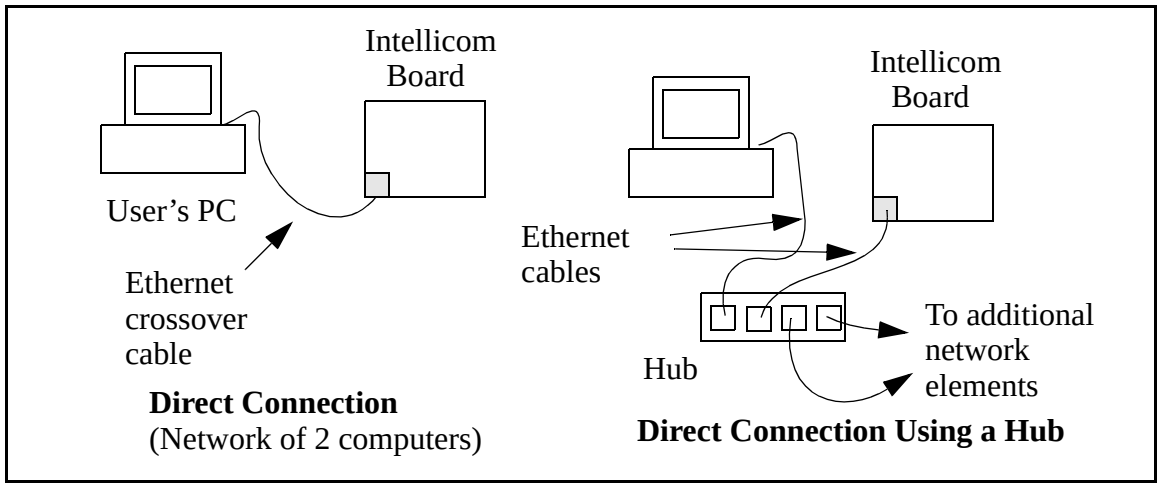
Obtaining IP addresses to interact over an existing, operating, network can involve a number of complications, and must usually be done with cooperation from your ISP and/or network systems administrator (if your company has one). For this reason, it is suggested that the user begin instead by using 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 virgin 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 Microsoft 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.

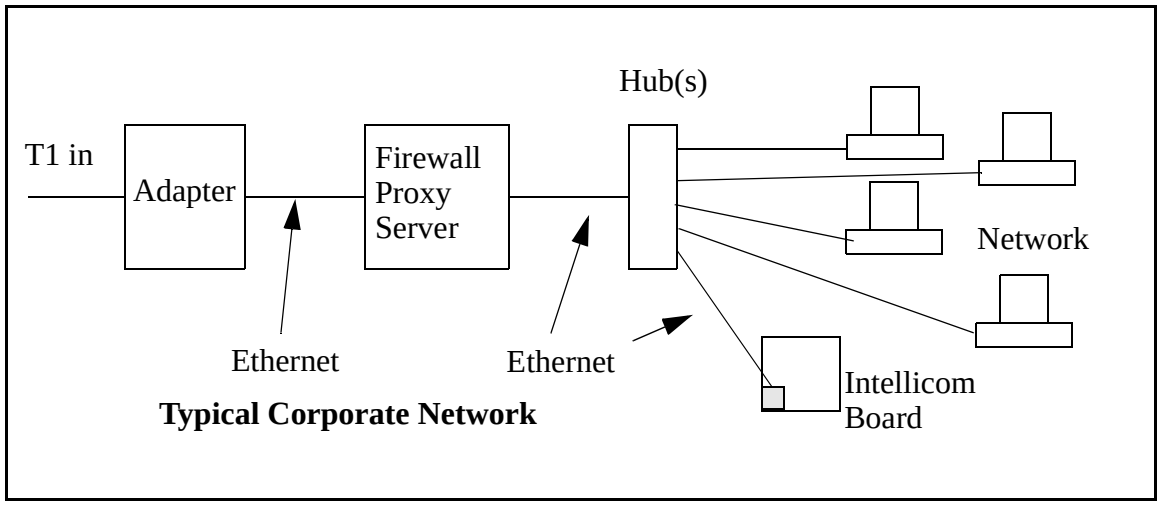
The following IP addresses are set aside for local networks and are not allowed on the Internet: 10.0.0.0 to 10.255.255.255, 172.16.0.0 to 172.31.255.255, and 192.168.0.0 to 192.168.255.255.

The Intellicom board uses a 10BaseT 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 unit that adapts to either 10 or 100 Mbps.



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. Y ou 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 IP Addresses Explained

IP (Internet Protocol) addresses are expressed as 4 decimal numbers separated by periods, for example:

216.103.126.155

10.1.1.6

Each decimal number must be between 0 and 255. The total IP address is a 32-bit number consisting of the 4 bytes expressed as shown above. A local network uses a group of adjacent IP addresses. There are always 2^N IP addresses in a local network. The netmask (also called subnet mask) determines how many IP addresses belong to the local network. The netmask is also a 32-bit address expressed in the same form as the IP address. An example netmask is:

255.255.255.0

This netmask has 8 zero bits in the least significant portion, and this means that 2^8 addresses are a part of the local network. Applied to the IP address above (216.103.126.155), this netmask would indicate that the following IP addresses belong to the local network:

216.103.126.0

216.103.126.1

216.103.126.2

etc.

216.103.126.254

216.103.126.255

The lowest and highest address are reserved for special purposes. The lowest address (216.102.126.0) is used to identify the local network. The highest address (216.102.126.255) is used as a broadcast address. Usually one other address is used for the address of the gateway out of the network. This leaves 256 - 3 = 253 available IP addresses for the example given.

5.4 How IP Addresses are Used

The actual hardware connection via an Ethernet uses Ethernet adapter addresses (also called MAC addresses.) These are 48-bit addresses and are unique for every Ethernet adapter manufactured. In order to send a packet to another computer, given the IP address of the other computer, it is first determined if the packet needs to be sent directly to the other computer or to the gateway. In either case, there is an IP address on the local network to which the packet must be sent. A table is maintained to allow the protocol driver to determine the MAC address corresponding to a particular IP address. If the table is empty, the MAC address is determined by sending an Ethernet broadcast packet to all

devices on the local network asking the device with the desired IP address to answer with its MAC address. In this way, the table entry can be filled in. If no device answers, then the device is nonexistent or inoperative, and the packet cannot be sent.

IP addresses are arbitrary and can be allocated as desired provided that they don't conflict with other IP addresses. However, if they are to be used with the Internet, then they must be numbers that are assigned to your connection by proper authorities, generally by delegation via your service provider.

5.5 Dynamically Assigned Internet Addresses

In many instances, IP addresses are assigned temporarily. This the normal procedure when you use a dial-up Internet service provider (ISP). Your system will be provided with an IP address that it can use to send and receive packets. This IP address will only be valid for the duration of the call, and further may not actually be a real Internet IP address. Such an address works for browsing the Web, but cannot be used for transactions originating elsewhere since no other system has any way to know the Internet address except by first receiving a packet from you. (If you want to find the IP address assigned by a dial-up ISP, run the program **`winipcfg`** while connected and look at the address for the ppp adapter under Windows 98.)

In a typical corporate network that is isolated from the Internet by a firewall and/or proxy server using address translation, the IP addresses are not usually actual Internet addresses and may be assigned statically or dynamically. If they are assigned statically, you only have to get an unused IP address and assign it to the Intellicom board. If the IP addresses are assigned dynamically, they you will have to get an IP address that is valid but outside of the range of IP addresses that are assigned dynamically. This will enable you to communicate from a PC on the network to the Intellicom board. If you want to communicate to the Intellicom board from the external Internet, then an actual Internet IP address must be assigned to the Intellicom board. It may be possible to setup the firewall to pass a real IP address, or it may be necessary to connect the Intellicom board in front of the firewall to accomplish this.

5.6 How to Set IP Addresses in the Sample Programs

Most of the sample programs such as shown in the example below use macros to define the IP address assigned to the board and the IP address of the gateway, if there is a gateway.

```
#define MY_IP_ADDRESS "216.112.116.155"
#define MY_NETMASK "255.255.255.248"
#define MY_GATEWAY "216.112.116.153"
```

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

```
#define MY_IP_ADDRESS "10.1.1.2"
#define MY_NETMASK "255.255.255.248"
// #define MY_GATEWAY "216.112.116.153"
```

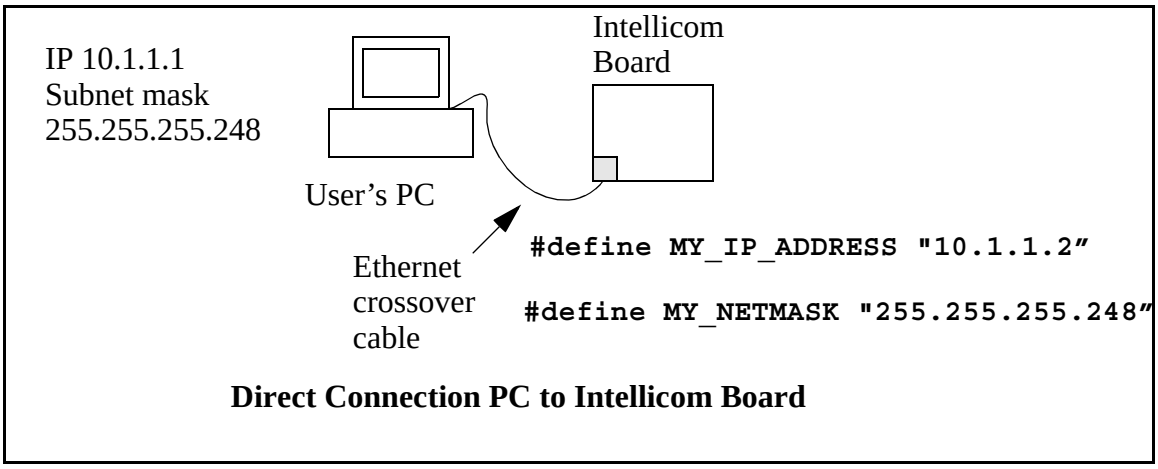
In this case, the gateway is not used and is commented out. The IP address of the board is defined to be 10.1.1.2. The IP address of you PC can be defined as 10.1.1.1.

5.7 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 10.1.1.1 with the subnetmask 255.255.255.248 under Windows 98, do the following.

Click on **Start > Settings > Control Panel** to bring up the Control Panel, and then double-click the Network icon. In the window find the line of the form **TCP/IP > Ethernet adapter name**. Double-click on this line to bring up the TCP/IP properties dialog box. You can edit the IP address directly and the subnet mask. (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.

The method of setting the IP address may differ for different versions of Windows, such as 95, NT or 2000.



Direct Connection PC to Intellicom Board

5.8 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.

5.9 Running More Demo Programs With Direct Connect

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.

5.10 Where Do I Go From Here?

If there are any problems at this point, call Z-World Technical Support at (530)757-3737.

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

Additional sample programs are described in the ***TCP/IP High-Level Protocols*** manual

Please refer to the ***TCP/IP High-Level Protocols*** manual and the ***TCP/IP Function Reference*** manual to develop your own applications. ***An Introduction to TCP/IP*** provides background information on TCP/IP, and is available on Z-World's Web site.

APPENDIX A.
INTELLICOM SPECIFICATIONS

A.1 Electrical and Mechanical Specifications

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

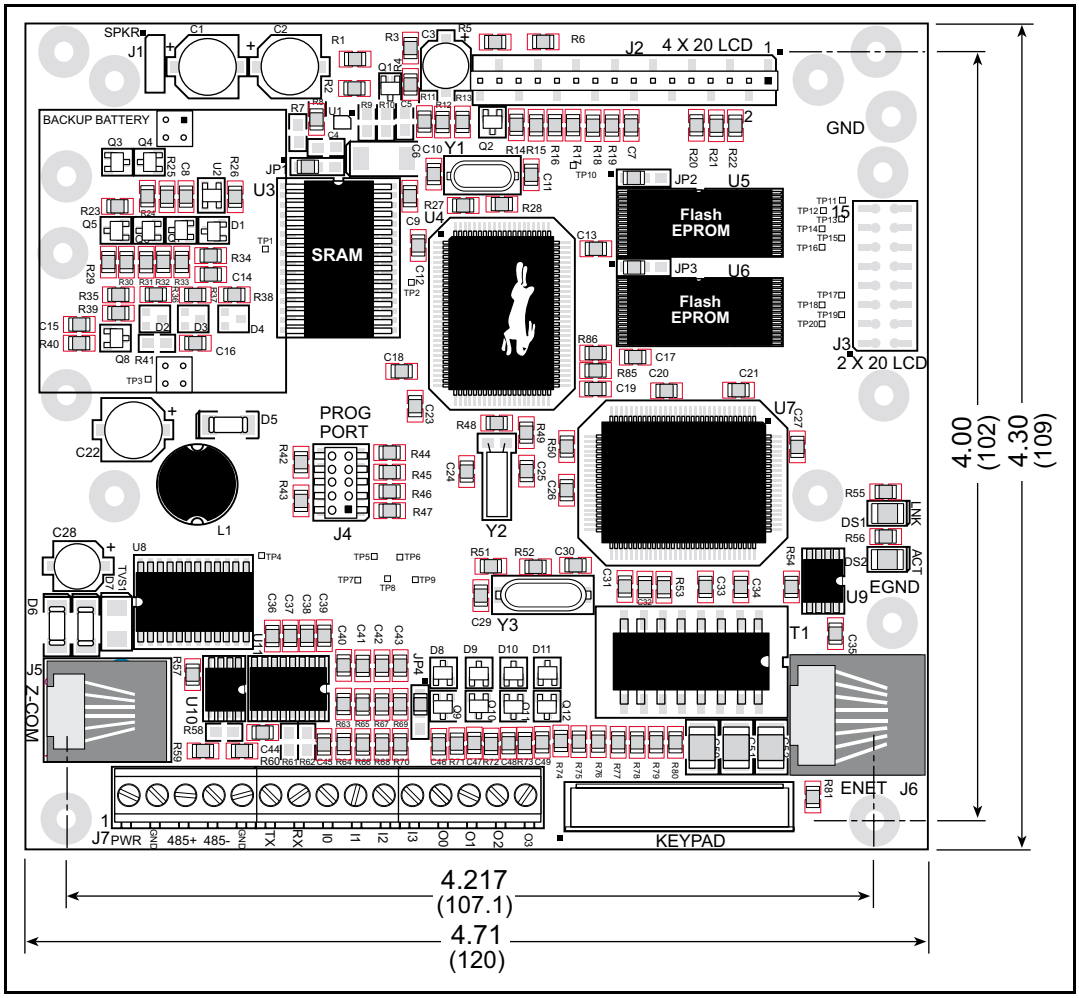


Figure A-1. Intellicom Board Dimensions

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

Table A-1. Intellicom Board Specifications

Parameter	Specification
Board Size (with backup battery board)	4.30" × 4.71" × 0.80" (109 mm × 120 mm × 20.3 mm)
Enclosure Size	5.7" × 6.7" × 2.0" (145 mm × 170 mm × 51 mm)
Display	Supertwist 4 × 20 LCD with backlighting
Keypad	2 × 6 domed tactile keypad with customizable legend
Connectors	15 screw terminals, 1 RJ-12, and 1 RJ-45
Operating Temperature	0°C to +50°C
Storage Temperature	–20°C to +60°C
Humidity	5% to 95%, noncondensing
Input Voltage	9 V to 40 V DC
Current	100 mA @ 24 VDC typical (backlighting on)
Ethernet Interface	Direct connection to 10BaseT Ethernet networks via RJ-45 connection
Digital Inputs	4 protected, 0 V to 5 V DC (protection from –36 V to + 36 VDC max.)
Digital Outputs	4 open collector, sinking (200 mA, 40 V DC max.)
Speaker Output	Software-adjustable volume and frequency
Microprocessor	Rabbit 2000™
Clock	18.432 MHz
SRAM	128K, surface mount (supports 32K–512K)
Flash EPROM	256K for program and data plus 256K for file storage (supports 128K–512K)
Timers	7 eight-bit timers available
Serial Ports	• 1 RS-232 (3-wire), 1 RS-485, and 1 RS-232 programming port • RS-232 (3-wire) and RS-485 may be reconfigured for 1 RS-232 (5-wire) or 2 RS-232 (3-wire)

Table A-1. Intellicom Board Specifications (continued)

Parameter	Specification
Serial Rate	Maximum asynchronous 115,200 bps for both serial ports
Watchdog/Supervisor	Yes
Time/Date Clock	Yes
Backup Battery	On backup battery board, 1000 mA·h, BR2477A

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.

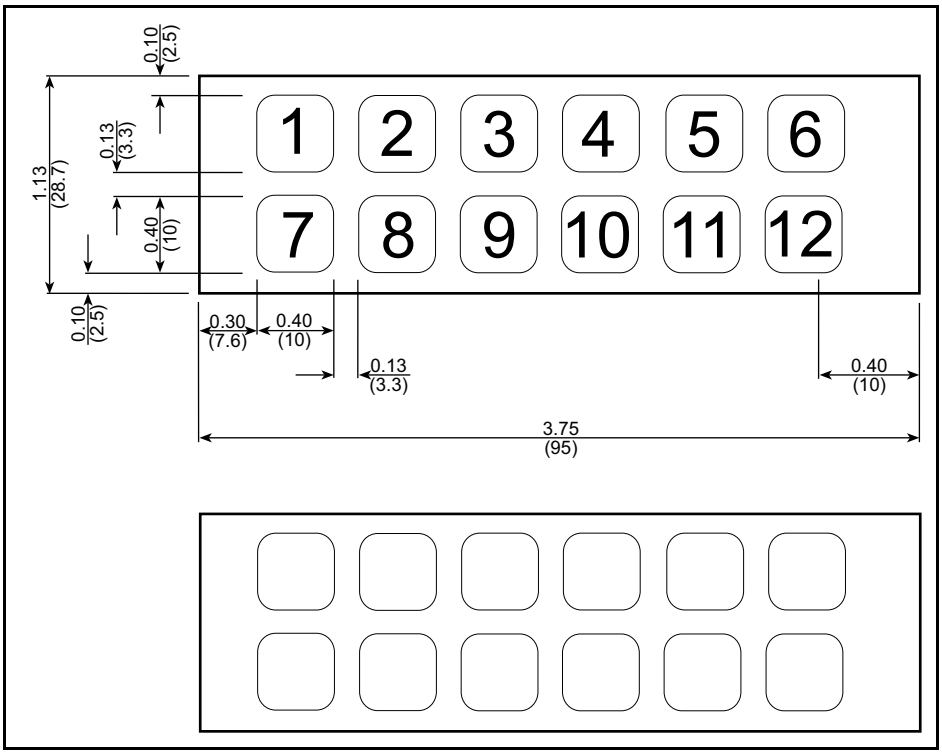


Figure B-1. Intellicom 2 × 6 Keypad Template

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.

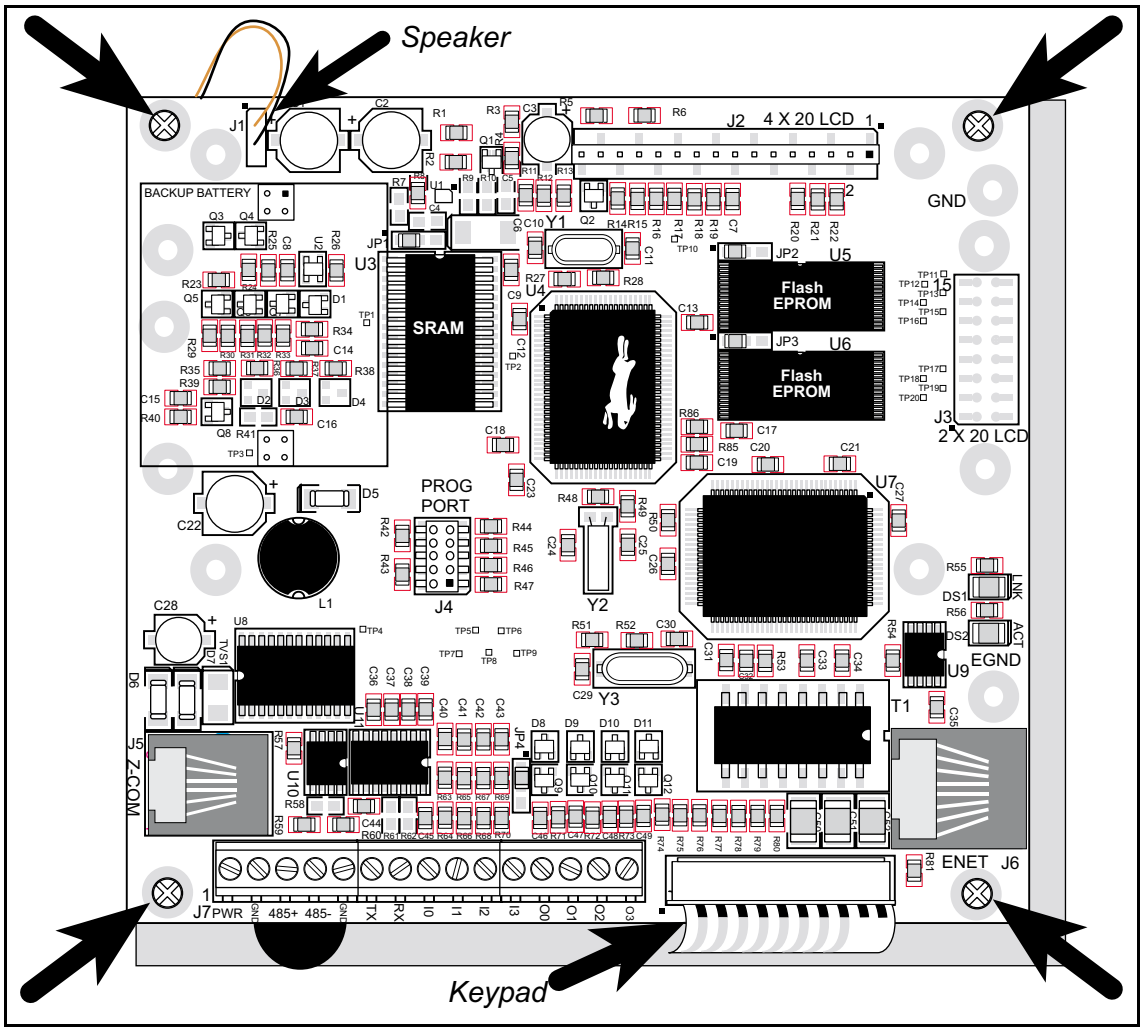


Figure B-2. Removing Intellicom Board from Front Panel

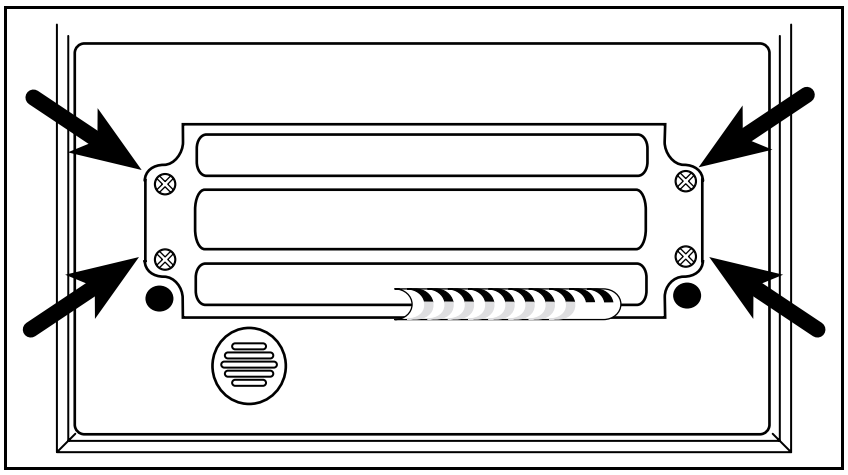


Figure B-3. Removing Keypad from Front Panel

B.2 Plastic Enclosure

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

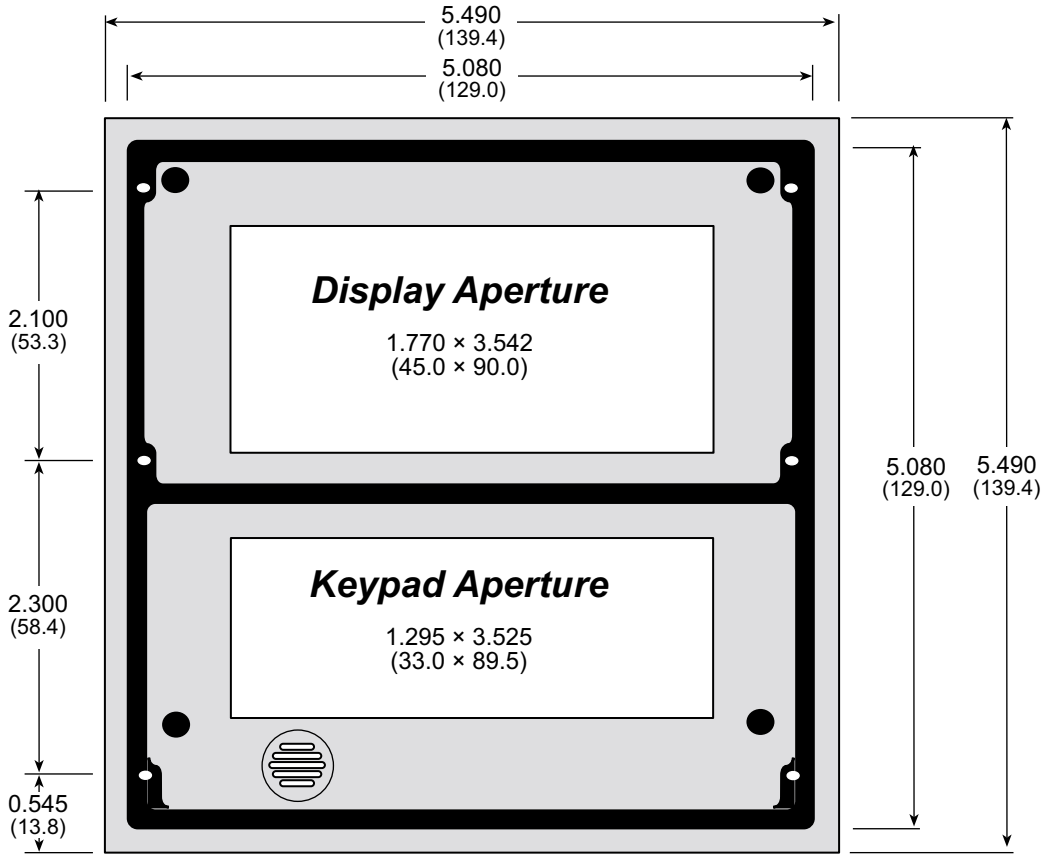


Figure B-4. Intellicom Front Panel Bezel Dimensions

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

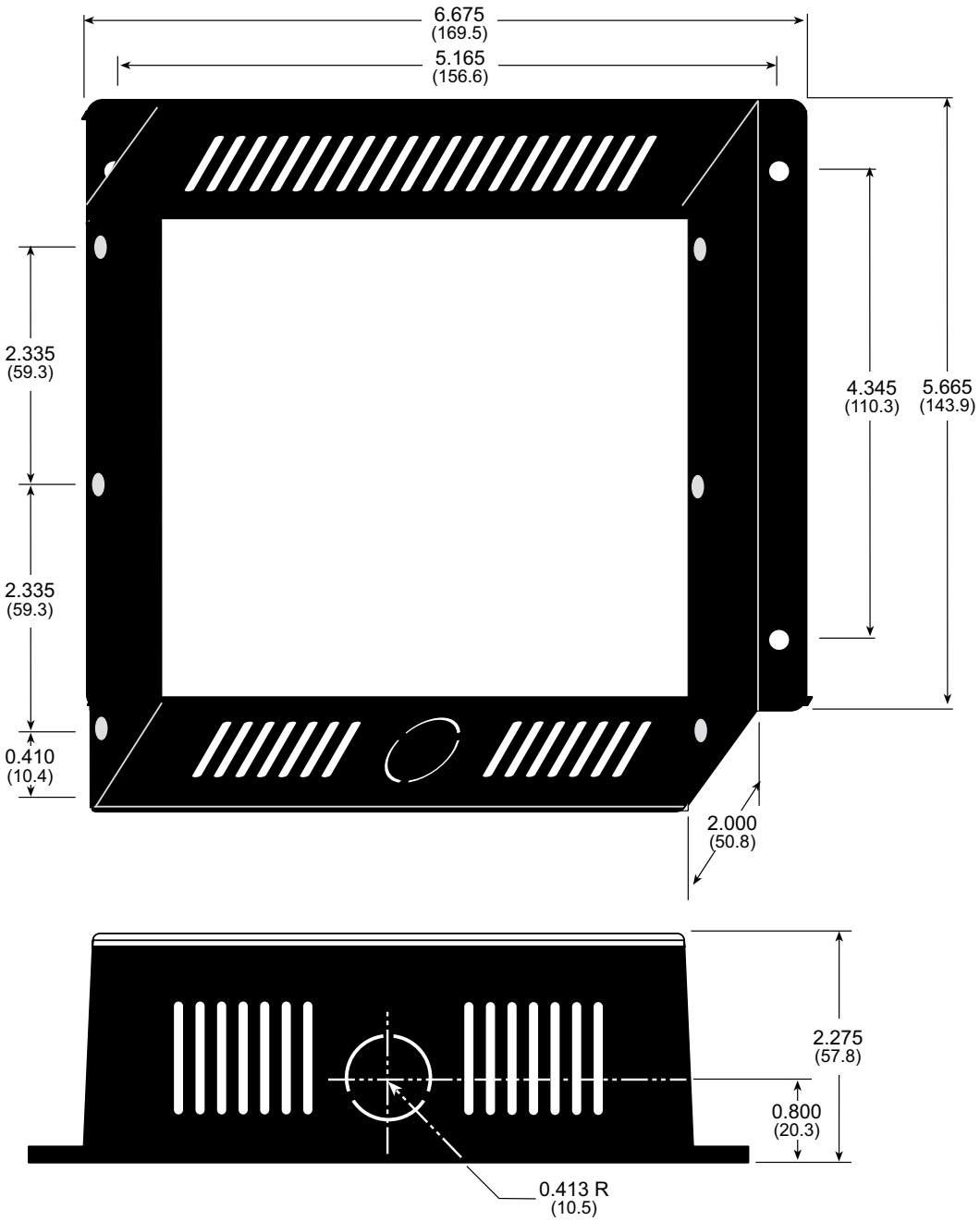


Figure B-5. Intellicom Outer Casing Dimensions

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 water-resistant unit by using the gasket supplied with the Intellicom to form a water-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.

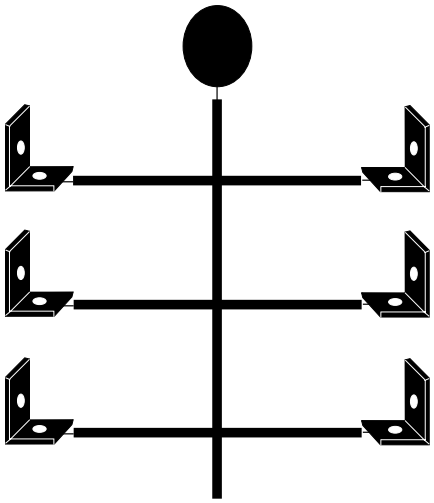


Figure B-6. Mounting Brackets and Speaker Plug

A watertight 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 watertight seal. Replace the speaker—a dab of glue around the edge of the speaker will hold it in place.

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

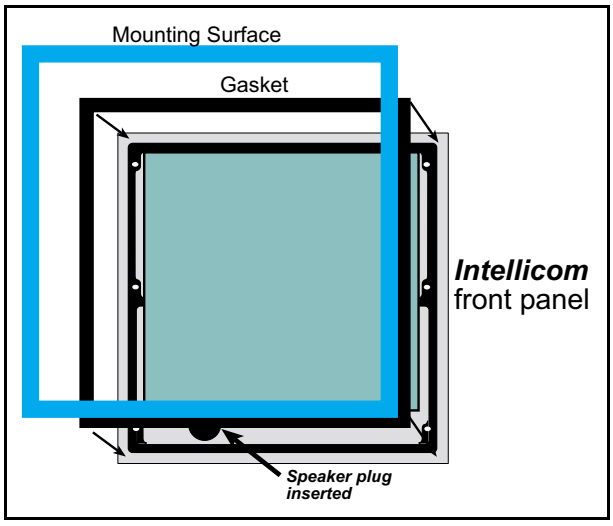


Figure B-7. Watertight Mounting of Intellicom

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

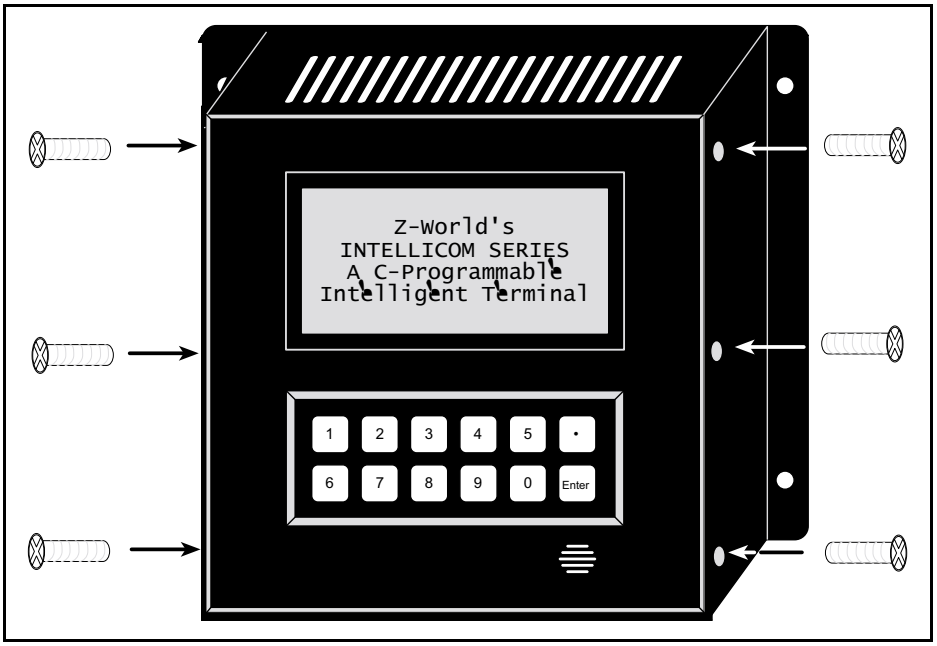


Figure B-8. Mounting Intellicom in Plastic Outer Casing

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.

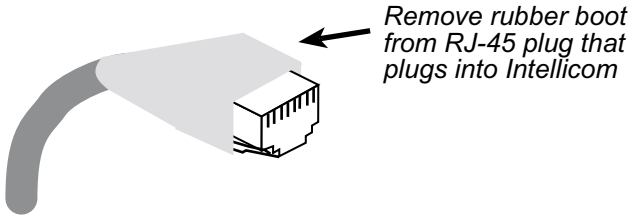


Figure B-9. Remove Rubber Boot from RJ-45 Connection to Intellicom

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 knock-out 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, Z-World 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.

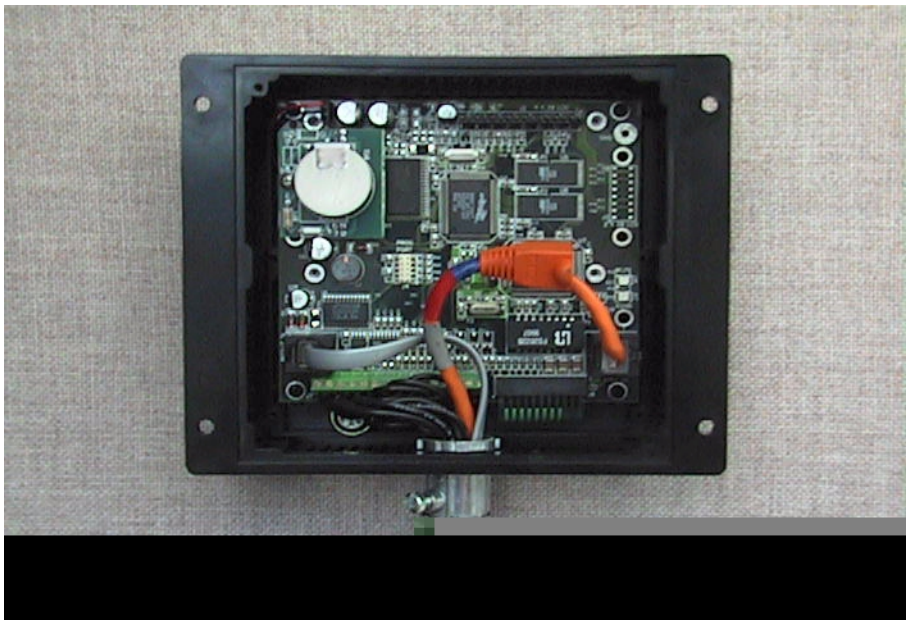


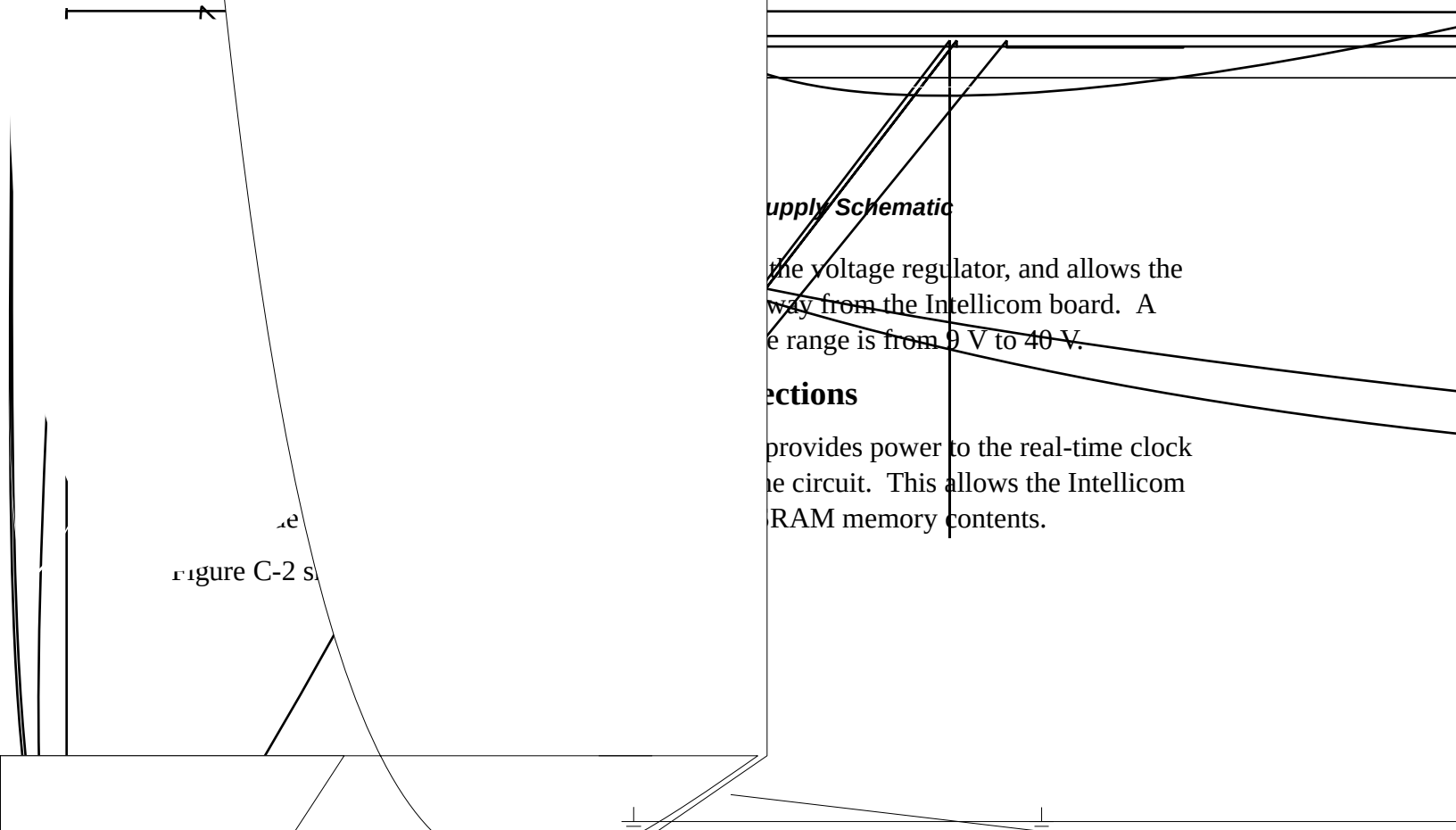
Figure B-10. Rear View of Intellicom with Outer Casing Showing Wiring from Conduit

APPENDIX C. POWER MANAGEMENT

C

P ternal source either through header J7
or -12 jack.

Th se polarity by Shottky diodes at D6
an has a low forward voltage drop, 0.3 V,
wh he Intellicom lower than a normal sil-
ico



upply Schematic

the voltage regulator, and allows the
way from the Intellicom board. A
e range is from 9 V to 40 V.

ections

provides power to the real-time clock
e circuit. This allows the Intellicom
RAM memory contents.

e

Figure C-2 s

C-2. Intellicom Backup Battery Board

.4(typuit. ll)7.6(ar)-7.lwerlntn on tAM whe202.5(-6.4µ(e)-6.4A)6(3 R)6.6(AM whe)-6.4()3.

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

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 Intellicom battery backup circuitry on the Intellicom board.

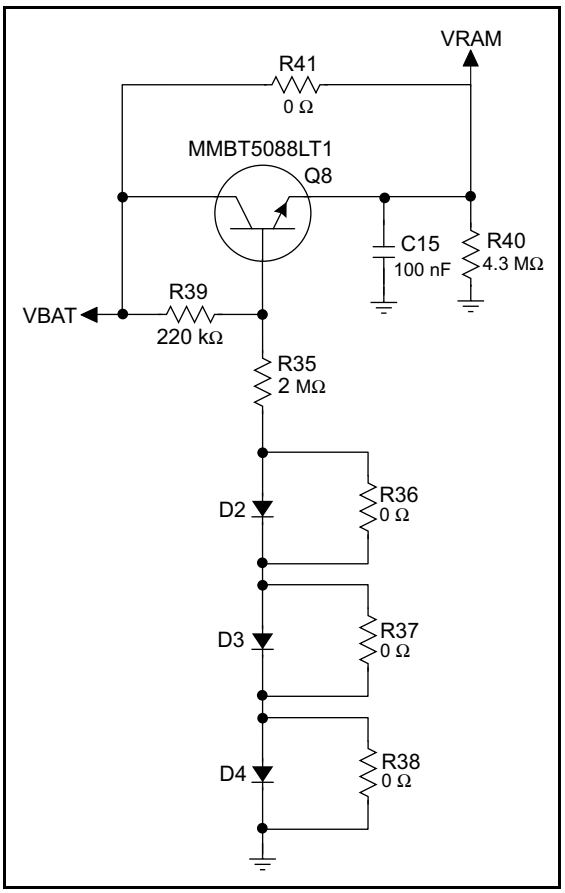


Figure C-3. Intellicom Battery Backup Circuit

Resistor R41, shown in Figure C-3, is typically not stuffed on the Intellicom board. VRAM and Vcc are nearly equal (<100 mV, typically 10 mV) when power is supplied to the Intellicom board. R14 on the backup battery board prevents any catastrophic failure of Q8 by limiting current from the battery.

Resistors R35 and R39 make up a voltage divider between the battery voltage and the temperature-compensation voltage at the anode of diode D2. This voltage divider biases the base of Q8 to about 0.9 × VBAT. V_{BE} on Q8 is about 0.55 V. Therefore, VRAM is about 0.9 × VBAT - 0.55 V, or about 2.15 V for a 3 V battery.

These voltages vary with temperature. VRAM varies the least because temperature-compensation diodes D2–D4 will offset the variation with temperature of Q8's V_{BE}. R36–R38 may be stuffed instead of the corresponding D2–D4 to provide the optimum temperature compensation.

Resistor R40 provides a minimum load to the regulator circuit.

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.

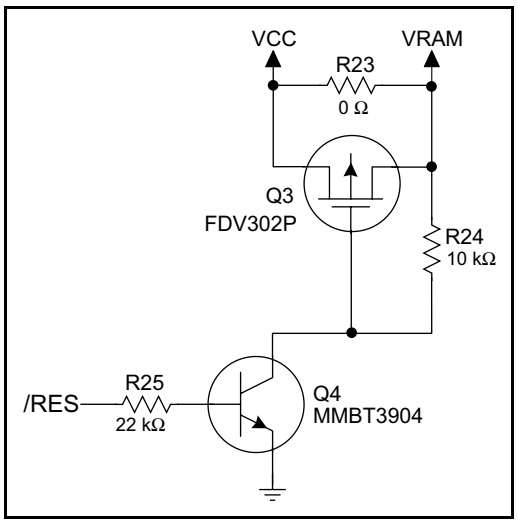


Figure C-4. VRAM Switch

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.

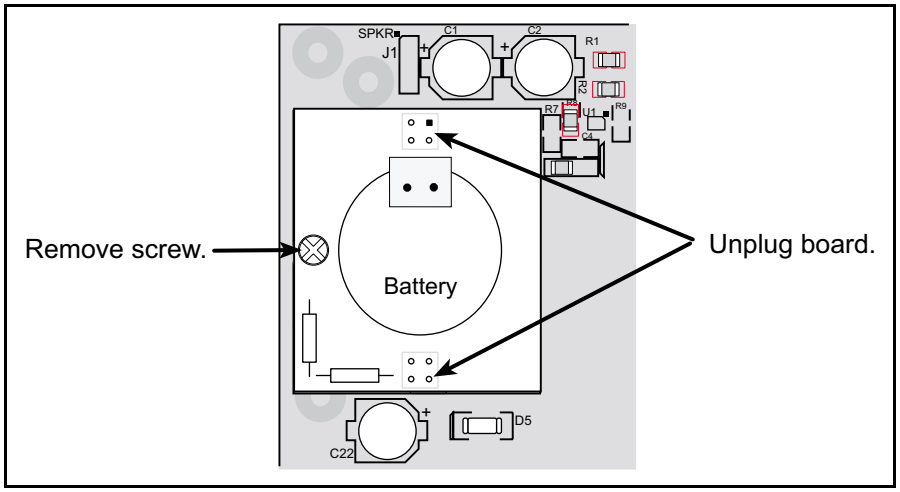


Figure C-5. Replacing Backup Battery Board

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 Z-World 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.

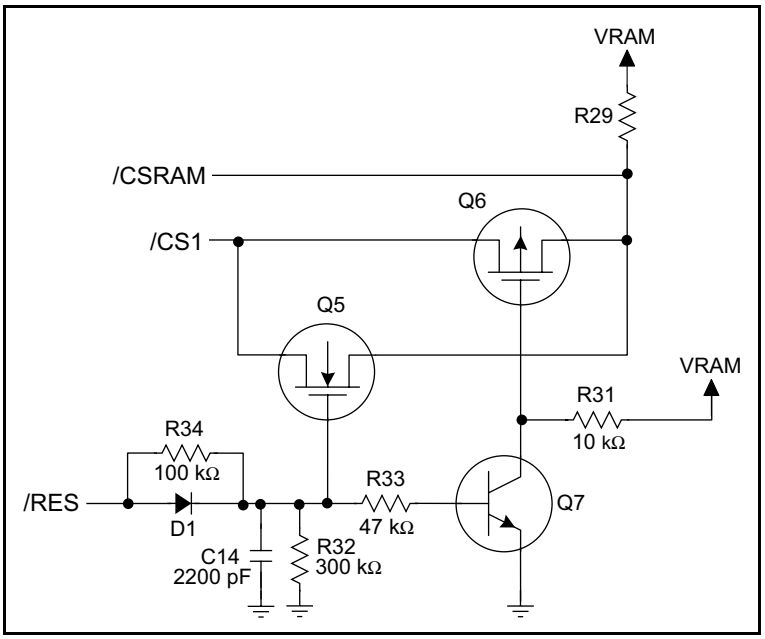


Figure C-6. Chip Select Circuit

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.

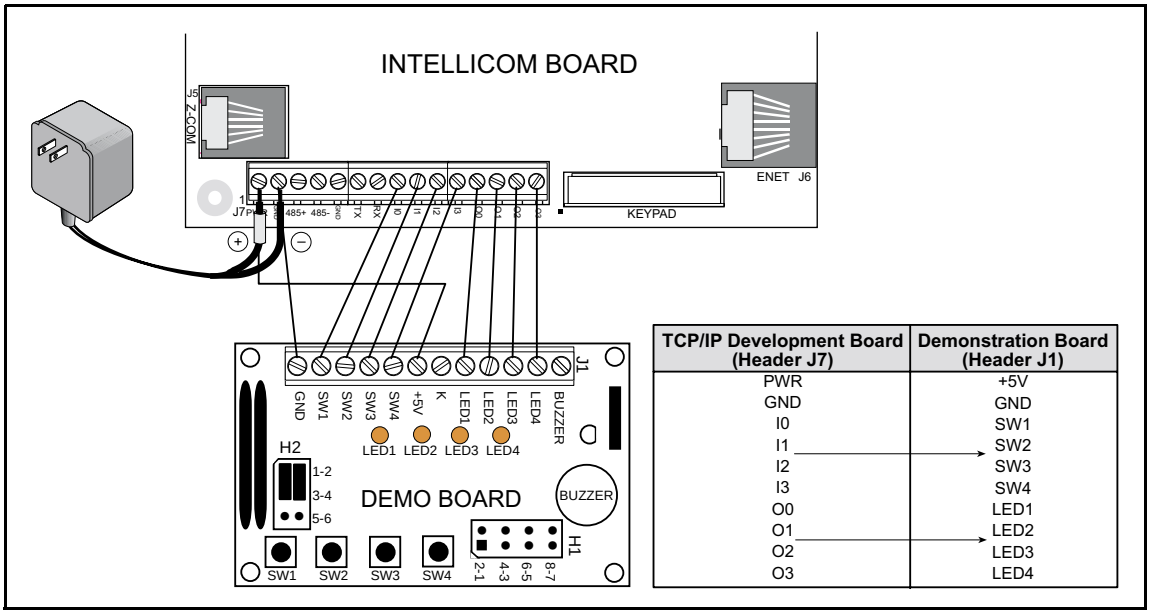
APPENDIX D. RUNNING SAMPLE PROGRAMS

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 Development 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.

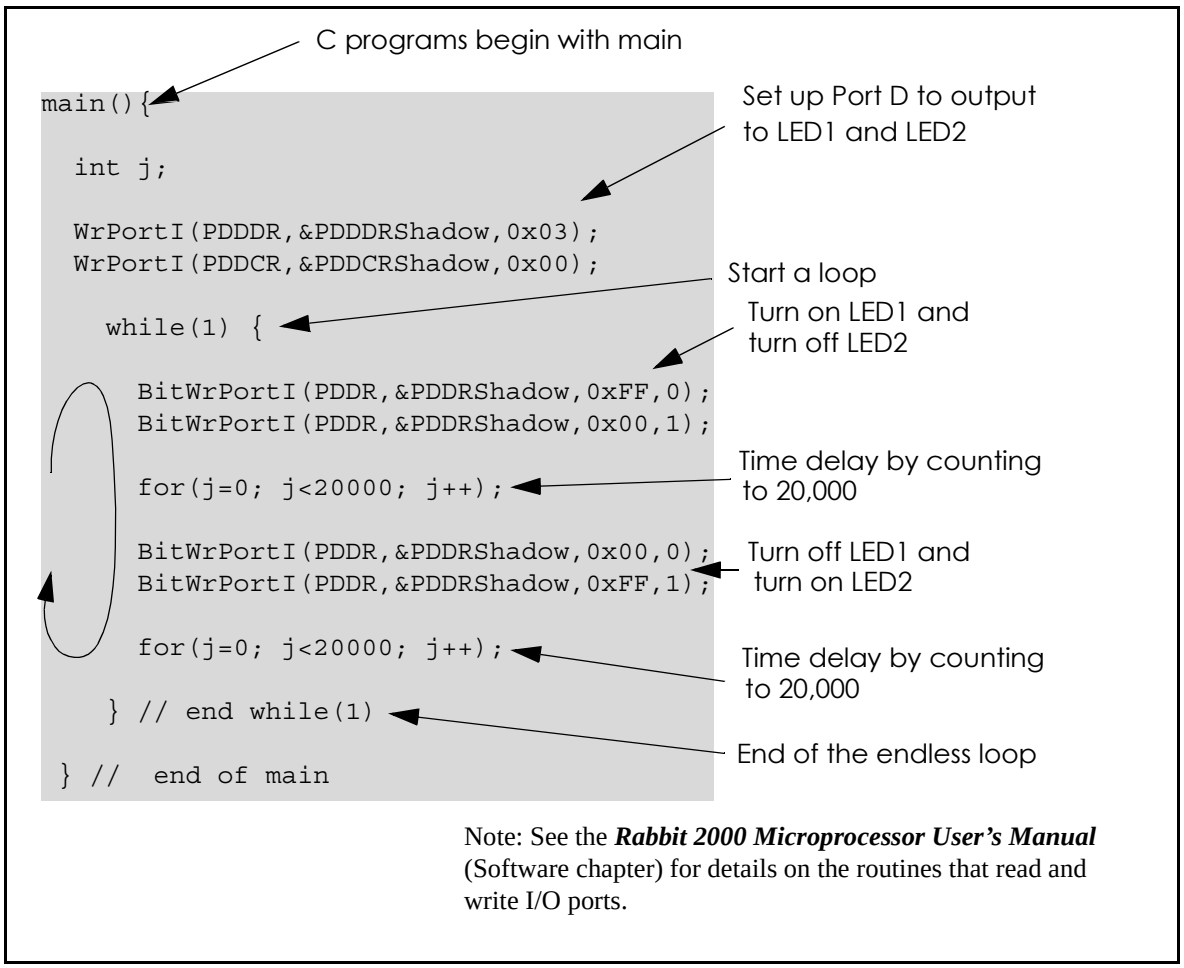


Figure D-2. Sample Program DEMOBRD1.C

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 **`rundwatch()`** 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 Premier 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 Premier 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
        BigLoopTop();                   // begin a big 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 function **`BigLoopTop()`** is used to collect some operations that are helpful to do once on every pass through the loop. Place the cursor on this function name **`BigLoopTop()`** and hit **<ctrl-H>** to learn more.

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 Premier 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 Premier User's Manual***.

Index

R

S

T

V

W

SCHEMATICS