Copyright 1993 - Remote Processing Corporation. All rights reserved. However, any part of this document may be reproduced with Remote Processing cited as the source.

The contents of this manual and the specifications herein may change without notice.

## TRADEMARKS

CAMBASIC™ and PC SmartLINK™ are trademarks of Octagon Systems Corporation.

Microsoft® BASIC is a trademark of Microsoft Corporation.

## NOTICE TO USER

The information contained in this manual is believed to be correct. However, Remote Processing assumes no responsibility for any of the circuits described herein, conveys no license under any patent or other right, and make no representations that the circuits are free from patent infringement. Remote Processing makes no representation or warranty that such applications will be suitable for the use specified without further testing or modification. The user must make the final determination as to fitness for a particular use.

Remote Processing Corporation's general policy does not recommend the use of its products in life support or in applications where the failure or malfunction of the board may threaten life or cause injury. Install redundant or backup safety systems as appropriate to the application.

Remote Processing Corporation
7975 E. Harvard Ave.
Denver, CO 80231
Ph: (303) 690 1588
Fax: (303) 690 1875
www.rp3.com

P/N 1193
Revision: 1.1

# TABLE OF CONTENTS


Chapter 1 Overview
- DESCRIPTION . . . . . 1
- MANUAL ORGANIZATION . . . . . 1
- MANUAL CONVENTIONS . . . . . 1
  - Symbols and Terminology . . . . . 1
- TECHNICAL SUPPORT . . . . . 2

Chapter 2 Setup and Operation
- INTRODUCTION . . . . . 3
- OPERATING PRECAUTION . . . . . 3
- EQUIPMENT . . . . . 3
- FIRST TIME OPERATION . . . . . 4
  - Using a PC . . . . . 4
  - Using a Terminal . . . . . 4
  - PC or Terminal . . . . . 4
- UPLOADING AND DOWNLOADING PROGRAMS . . . . . 4
  - Uploading programs . . . . . 4
  - Downloading programs . . . . . 5
  - Other communications software . . . . . 5
  - Editing programs and programming hints . . 5
- WHERE TO GO FROM HERE . . . . . 6
- TROUBLESHOOTING . . . . . 7

Chapter 3 Saving Programs
- INTRODUCTION . . . . . 8
- SAVING A PROGRAM . . . . . 8
- AUTORUNNING . . . . . 8
- PREVENTING AUTORUN . . . . . 8
- LOADING A PROGRAM . . . . . 9
- SAVING DATA TO EPROM . . . . . 9
  - COM1 SERIAL PORT . . . . . 10
  - COM2 SERIAL PORT . . . . . 10
- ACCESSING SERIAL BUFFERS . . . . . 10
- SERIAL PORT FILE NUMBERS . . . . . 11
- COMMANDS . . . . . 11
- SERIAL CABLE PIN OUT . . . . . 11

Chapter 5 Data Memory
- INTRODUCTION . . . . . 12
- BATTERY BACKUP . . . . . 12
- STORING VARIABLES IN RAM . . . . . 12
- ASSEMBLY LANGUAGE INTERFACE . . . 13
- COMMANDS . . . . . 13

Chapter 6 Digital Lines
- INTRODUCTION . . . . . 14
- DIGITAL I/O PORT . . . . . 14
  - Pull up resistors . . . . . 14
  - High current output . . . . . 14
  - Interfacing to an opto-module rack . . . . . 15
  - Interfacing to switches and other devices . 16
  - Configuring digital I/O lines . . . . . 16
  - Digital I/O programming example . . . . . 16
  - Connector pin out - J2 . . . . . 16
  - Connector pin out - J3 . . . . . 17
- COMMANDS . . . . . 17

Chapter 7 Calendar/Clock
- DESCRIPTION . . . . . 18
- INSTALLATION . . . . . 18
- SETTING DATE AND TIME . . . . . 18
- COMMANDS . . . . . 18

Chapter 8 Keypad Port
- INTRODUCTION . . . . . 19
- PROGRAMMING EXAMPLE . . . . . 19
- KEYPAD PORT PIN OUT - J5 . . . . . 19
- COMMANDS . . . . . 20

Chapter 9 Speaker
- DESCRIPTION . . . . . 21
  - Connecting a speaker . . . . . 21
- SYNTAX . . . . . 21
  - Programming example . . . . . 21

Technical Information
- ELECTRICAL . . . . . 22
- MECHANICAL . . . . . 22
- MEMORY AND I/O MAP . . . . . 22
- JUMPER DESCRIPTIONS . . . . . 22
  - Serial Port pin out . . . . . 23

# CHAPTER 1 OVERVIEW

## DESCRIPTION

The RPC-150 is an embedded controller with a built in Basic language. Several features make it suitable as a stand alone unit:

Built in CAMBASIC programming language autoruns at power up. On card EPROM programmer saves programs to 30K.

Keypad port for operator interface. The 16 position keypad is automatically scanned and is read using the KEYPAD command.

Two RS-232 serial ports are programmable for baud rate, parity, length, and stop bits. Both inputs and outputs have a 256 byte buffer.

47 general purpose digital I/O lines, 7 of which are high current outputs. These lines can connect to another opto rack.

Built in EPROM programmer and CAMBASIC software save programs for autorun on power up or reset.

The RPC-150 uses a 64180 CPU operating at 9 Mhz. It operates stand alone or on a network using RS-485 adapter. Its 4.5" x 6" size makes it easy to mount in a NEMA box.

CAMBASIC programming language is standard. This language was adapted for the RPC-150 for control and data acquisition applications. A complete description of CAMBASIC commands is in the *CAMBASIC Programming Guide*.

Program development can take place on your PC, using your word processor, or on the RPC-150. Programs from your PC can be downloaded using PC SmartLINK or other serial communication program.

## MANUAL ORGANIZATION

This manual provides all the information required to install, configure, and use the features on the RPC-150.

This manual assumes you are familiar with some type of BASIC programming software. The syntax used by CAMBASIC is similar to Microsoft's GW or QuickBASIC. If you are not experienced with BASIC software, you may want to refer to books and training programs available through your local software store. The *CAMBASIC Programming Manual* has information and examples for all commands.

***NOTE:*** The RPC-150 uses a Hitachi Z180 processor. Additional information can be obtained from Hitachi or a local representative. Order hardware manual #U77, software manual #U92.

## MANUAL CONVENTIONS

Information appearing on your screen is shown in a different type.

Example:

```
CAMBASIC (tm) (c) 1985-93
Octagon Systems Corporation
Remote Processing Corporation
All rights reserved
Bytes Free - 27434
```

**Symbols and Terminology**

***NOTE:*** Text under this heading is helpful information. It is intended to act as a reminder of some interaction with another part of the manual or device that may not be obvious.

***WARNING:***

Information under this heading warns you of situations which might cause catastrophic or irreversible damage.

W[-] Denotes jumper block pins.

< xxx> Paired angle brackets are used to indicate a specific key on your keyboard. For example < esc> means the escape key.

BASIC uses the decimal convention for designating addresses and data. There are times, however, when hexadecimal notation is more convenient to use. The hexadecimal notation used in this manual and by CAMBASIC is the ampersand character (&) before the number. A &8C stands for 8C hexadecimal.

## TECHNICAL SUPPORT

If you have a question about the RPC-150 or CAMBASIC used on it and can't find it in this manual, call us and ask for technical support.

When you call, please have your RPC-150 and CAMBASIC manuals ready. Sometimes it is helpful to know what the RPC-150 is used for, so please be ready to describe its application as well as the problem.

Phone: 303-690-1588
FAX: 303-690-1875

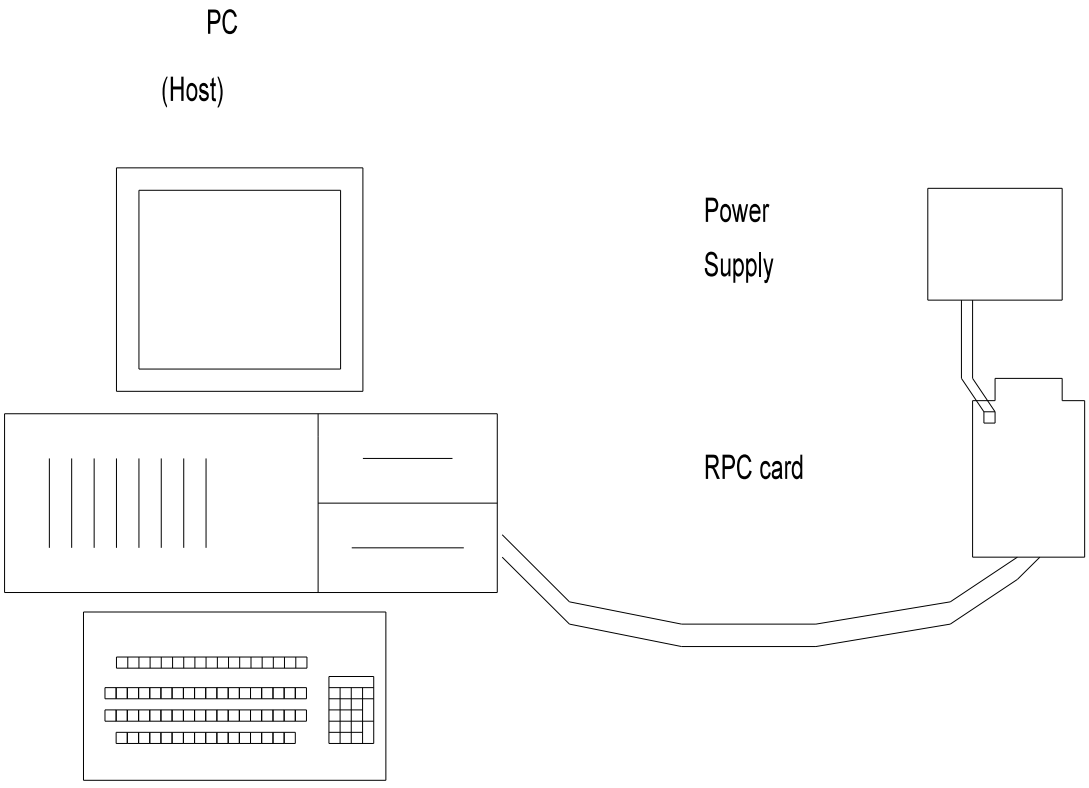



**Figure 1-1 System layout**

## INTRODUCTION

The RPC-150 is ready to program as soon as you connect it to a terminal or PC and apply power. This chapter describes what is needed to get a sign- on message and begin programming.

Requirements for uploading and downloading programs is discussed. A "Where to go from here" section directs you to the chapters to read in order to use the various capabilities of the RPC-150. Finally, a troubleshooting section helps out on the most common problems.

## OPERATING PRECAUTION

The RPC-150 is designed to handle a wide variety of temperature ranges and operating conditions. These characteristics require using CMOS components. CMOS is static sensitive. To avoid damaging these components, observe the following precautions before handling the RPC-150.

1. Ground yourself before handling the RPC-150 or plugging in cables. Static electricity can easily arc through cables and to the card. Simply touching a metal part on your PC can greatly reduce the amount of static.

2. Do not insert or remove components when power is applied. While the card is a + 5 volt only system, other voltages are generated on the card. Applying them in the wrong sequence can destroy a component.

## EQUIPMENT

You will need the following equipment to begin using the RPC-150:

RPC-150 embedded controller
PC with a serial port and communications program (such as PC SmartLINK)
or
Terminal
VTC-10 serial cable
+ 5V, 300 ma power supply

The *CAMBASIC Programming Manual* is strongly recommended. Refer to *Chapter 4 Serial Ports* for wiring information to make your own cable.

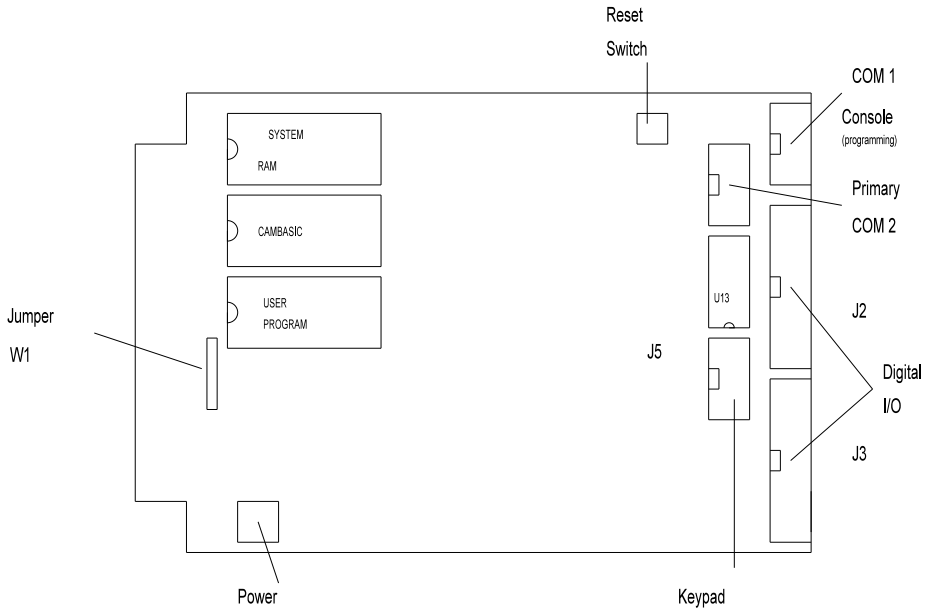



**Figure 2-1 RPC-150 Connector Layout**

## FIRST TIME OPERATION

Become familiar with the locations of the connectors before getting started. See Figure 2-1.

RPC-150 jumpers have been set at the factory to operate the system immediately. For first time operation, do not install any connectors or parts unless specified below. Jumpers should be kept in default positions.

1. The RPC-150 needs + 5 ±0.25 volts at less than 100 ma. Any well regulated supply that supplies this will work. Be careful when using "switching" power supplies. Some supplies do not regulate properly unless they are adequately loaded.

   The power supply should have a rise time to + 5 volts in under 100 ms. If it takes longer than this, you may have to manually reset the card. Most power supplies meet this requirement.

   Make sure power is off. Connect the power supply to the appropriately marked terminals on the RPC-150.

2. You can use either a PC or CRT terminal to program the RPC-150. Connect one end of the VTC-10 connector to then 10 pin COM1 (console) port on the RPC-150. Refer to Figure 2-1 for connector location.

   **Using a PC**
   Connect the VTC-10 serial cable to the PC's COM1 or COM2 port. You may need a 9 pin male to 25 pin female adapter. The VTC-10 is designed to plug directly into the 9 pin serial port connector on a PC.

   Start up your serial communication program (PC SmartLINK or other). Set communication parameters to 19.2K baud, 8 data bits, no parity, 1 stop.

   **Using a Terminal**
   Follow your terminal instructions to set the baud rate to 19.2K baud, 8 data bits, no parity, and 1 stop. You may need a 9 pin male to 25 pin female adapter to connect the VTC-10.

   **PC or Terminal**
   The RPC-150 does not send a CTS signal to the PC or terminal. If your terminal or communications software requires this or other signals (DCD, DSR), you may have to tie them to the appropriate levels. You may be able to ignore these lines in software.

3. Turn on your power supply. On power up a copyright message is printed.

   ```
   CAMBASIC (tm)  V1.00
   (c) 1985-94 Octagon Systems Corp
   (c) 1994 Remote Processing Corp
   All rights reserved - Free - 29434
   ```

   If a nonsense message appears, your terminal or PC may not be set to the appropriate communication parameters. If the system still does not respond, refer to *TROUBLESHOOTING* later in this chapter.

4. The system is now in the "immediate mode" and is ready for you to start programming. Type the following program (in upper or lower case:

   ```
   10 FOR X = 0 TO 2
   20 PRINT "   Hello ";
   30 NEXT
   40 PRINT
   ```

   Now type `RUN`

   The system will display:

   ```
   Hello Hello Hello
   ```

## UPLOADING AND DOWNLOADING PROGRAMS

Downloading programs means transferring them from your PC (or terminal) to RAM on the RPC-150. Uploading means transferring programs from RAM back to the PC. This section explains how to do both of these procedures using PC SmartLink. Generalized instructions for other terminal programs are given later.

**Uploading programs**

In the previous section, you wrote a test program. To upload that program to a PC and save it to disk:

1. Press the < F1> key. A window with the main menu will appear.

2. Press the letter U (upper or lower case). Your program will begin to transfer from RAM to the PC. When menu appears.

3. To save a program to disk, type the letter S. You are prompted for a file name. Enter the file name you want the program saved under.

4. Press < F2> to return to the immediate mode.

***NOTE:*** Some versions of PC SmartLINK have pull down menus or will operate differently. Refer to the SmartLINK manual for the version you are using.

**Downloading programs**

To practice downloading a program, type

```
NEW< return>
```

Perform the following when using PC SmartLINK:

1. Press the < F1> key to view the main menu.

2. SmartLINK has a buffer which is used to temporarily store the program. If you followed these instructions without exiting SmartLINK, the previously uploaded program is in the buffer and may be downloaded. However, lets assume you just started SmartLINK. Press the L key to get the program from the disk.

3. Enter the filename to get the file.

4. Press D to download the program.

5. Press the < F2> key to return to the program. You can list the program by typing:

```
list
```

or

```
/
```

**Other communications software**

The following is general information when using another terminal emulation program (Procomm, Windows Terminal, etc.).

When uploading or downloading files, select ASCII text format. XMODEM, YMODEM, or other formats are not used.

CAMBASIC does not know when you are typing in a program or if something else (laptop or mainframe) is sending it characters. The upload and download file does not contain any special control codes, it is simply ASCII characters.

Uploading programs is simply a process of receiving an ASCII file. You or your program simply needs to send "LIST" to receive the entire program. The default baud rate (19200) is rather high. Make sure your PC and communications software can work at these baud rates. PROCOMM was tested on a 12 Mhz 286 PC and it worked fine. Windows Terminal on the same PC had problems at much slower baud rates.

Downloading a program requires transmitting an ASCII file. CAMBASIC is an incremental line compiler. As you type in (or download) a line, CAMBASIC compiles that line. The time to compile a line depends upon its complexity and how many line of code have been entered.

CAMBASIC must finish compiling a line before starting the next one. When a line is compiled, a "> " character is sent by the card. This should be your terminal programs pacing character when downloading a program.

If your communications program cannot look for a pacing prompt, set it to delay transmission after each line is sent. A 100 ms delay is usually adequate, but your CAMBASIC program may be long and complex and require more time. A result of a short delay time is missing or garbled program lines.

CAMBASIC sends out escape sequences to clear the screen. This sequence may appear as < -; on your screen. Usually this is not a problem.

COM1 on the RPC-150 does not recognize the CTS or RTS lines. The CTS line is NOT pulled high on the RPC-150. The effect of not recognizing these lines is your PC or terminal cannot hold off the RPC-150's transmission. Converse, the RPC-150 cannot hold off the host from sending it data. Set your terminal program to not recognize or use the CTS and RTS lines.

**Editing programs and programming hints**

Files uploaded or downloaded are simply ASCII DOS text files. No special characters or control codes are used. You may create and edit programs using your favorite word processor or editor. Just be sure to save files in DOS text format.

A technique used to further program documentation and reduce code space is the use of comments in a

downloaded file. For example, you could have the following in a file written on your editor:

```
'Check temperature

'Read the output from the RTD and
' calculate the temperature

2200 a = ain(0) :'Get temp
```

The first 3 comments downloaded to the RPC-150 would be ignored. Similarly, the empty lines between comments are also ignored. Line 2200, with its comment, is a part of the program and could be listed. The major penalty by writing a program this way is increased download time.

Notice that you can write a program in lower case characters. CAMBASIC translates them to upper case.

Some programmers put "NEW" as the first line in the file. During debugging, it is common to insert "temporary" lines. Adding NEW ensures that these lines are gone. Downloading time is increased when the old program is still present.

If you like to write programs in separate modules, you can download them separately. Modules are assigned blocks of line numbers. Start up code might be from 1 to 999. Interrupt handling (keypad, serial ports) might be from lines 1000 to 1499. Display output might be from 1500 to 2500. The programmer must determine the number of lines required for each section.

When replacing a program or section, downloading time is increased. Blocks of line numbers cannot be renumbered by CAMBASIC when other parts of the program are installed. However, if a particular section is the only program downloaded, then line renumbering in that range is possible. Refer to the CAMBASIC RENUM command.

CAMBASIC automatically formats a line for minimum code space and increased readability. For example, you could download the following line of code:

```
10 fora=0to5
```

When you listed this line, it would appear as:

```
10 FOR A = 0 TO 5
```

Spaces are initially displayed but not stored. The following line:

```
10 for a = 0 to 5
```

would be compressed and displayed as in the second example above. Spaces are removed.

The *CAMBASIC Programming Manual* has more information about increasing program speed and editing options.

Instead of uploading and downloading programs, you can save them to the on card EPROM. This is useful if you are using a terminal to write programs. Make sure the 'AUTORUN' jumper is installed (See *Chapter 3 SAVING PROGRAMS*). To prevent automatic program execution on power up, insert the STOP statement at the beginning of the program (such as line 1). When you power up the RPC-150, the program is transferred into RAM and executed. Delete the program line with the STOP statement to normally start programs. When saving programs, be sure to reenter the STOP statement with its line number.

## WHERE TO GO FROM HERE

| If you want to do this: | Turn to Chapter |
|---|---|
| Save a program | 3 |
| Autorun a program | 3 |
| Know more about serial ports | 4 |
| Battery backing up RAM | 5 |
| Use RAM to save variables | 5 |
| Configure digital I/O lines | 6 |
| Get switch status | 6 |
| Use high current outputs | 6 |
| Connect an external opto rack | 6 |
| Installing calendar/clock module | 7 |
| Using a keypad | 8 |
| Speaker port | 9 |

Also, refer to the table of contents for a listing of other functions.

## TROUBLESHOOTING

You probably turned to this section because you could not get the sign on message. If you are getting a sign on message but can't enter characters, then read section 5 below. The following are troubleshooting hints:

1. Check the power source. Power is 5 ±0.25 volts. Make sure it is a clean 5 volt source. If it dips intermittently to about 4.5 volts (due to switching noise or ripple), the CPU may read an instruction incorrectly and consequently 'crash'.

   The power supply must be able to go from 0 to + 5 volts in under 100 ms. If it takes longer than this, the card may come out of reset too soon and the CPU will not operate correctly.

   The best way to reset a card is to push the reset button. When powering off then on, make sure the supply gets to at least 0.7 volts before turning on the supply. The reset circuit must discharge completely for a clean reset. This is a problem on large, linear supplies.

2. Check the COM1 port. COM1 is also known as J1. Remove the connector from COM1. Refer to the outline drawing earlier in this chapter. Connect an oscilloscope (preferred) or a voltmeter to pin 3 (Txd) and ground. Pin 3 should be -6 volts or more negative. (Pin 1 is designated by the ### symbol on the connector. Pin 3 is next to it, nearer the key opening.) If you have -6 volts or more, press the reset switch. If you have a scope attached, you should see a burst of activity. If you have a volt meter, you should see a change in voltage. Using a Fluke 8060A set to measure AC, you should see a momentary reading above 2 volts. Press reset several times to make sure it captures it.

3. Install the cable and make sure the voltages and output activity are still there. Output is from pin 3 on the VTC-10. Check to make sure something is not shorting the output.

4. Check the serial parameters on your PC or terminal. They should be set to:

   19200 baud, no parity, 8 data bits, 1 stop

5. Many times a bad or wrong serial cable can cause characters to get printed but you can't send anything to the card. Make sure you are using a VTC-10 cable. This cable is wired per *Chapter 4, SERIAL CABLE PIN OUT*.

If all of this fails, call technical support listed under *Chapter 1, TECHNICAL SUPPORT*.

## INTRODUCTION

Programs are stored in socket U3. An optional real time clock module, a DS-1216EM may also be installed in U3. See *Chapter 7* for calendar/clock installation and operation.

You can store one program up to a maximum size of about 28K bytes. A general rule to determine program storage requirements is one line requires 40 bytes. 28K bytes would store over 700 lines of code. Your application could be significantly more or less, depending upon the number of commands / line, comments, and print statements. Another indication of program size is to use the file length as saved on a PC disk.

An EPROM is non-volatile flash type technology, having an unlimited number of read cycles and a limited number of write cycles (about 10,000). A program is not run from EPROM. It is transferred to RAM and run from there. Programs in RAM are run and can be modified. They can be saved to EPROM for auto execution later.

The RPC-150 can be set to autorun on power up or reset by installing jumper (W1[17-18]). When autorun is on, the program in EPROM is loaded into RAM and begins to execute immediately.

The EPROM is write-protected with a software lock, so accidental writes on power-on or -off are almost impossible. You cannot disable the lock except when executing the SAVE command. To save parameters, you must use battery backed RAM and save data to U1.

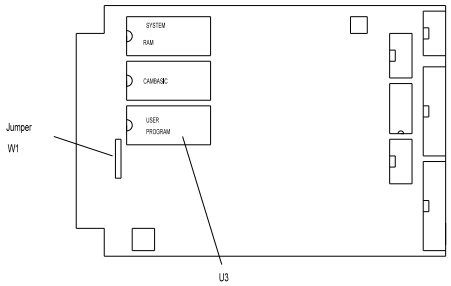



**Figure 3-1 Autorun jumper**

## SAVING A PROGRAM

To save a program, set jumper W1[3-4] and [17-18]. You may set the jumper even if the power is on. Remember to discharge any static electricity before installing or removing the jumper. For this example, assume you wanted to save the following program:

```
10 FOR N = 0 TO 2
20 PRINT "Hello ";
30 NEXT
40 PRINT
```

If this program is not already in, type it in now (or, if you prefer, use your own program).

Type in the following command:

```
SAVE
```

CAMBASIC will compile the program, program the EPROM, and verify its contents.

```
Compile...Write...Verify
```

The time it takes to do all of this depends upon the length and complexity of the program. Generally, it will be from 1 to 20 seconds. The ready prompt (> ) will appear when the program has been successfully saved to the EPROM. If the program does not write to the EPROM, an error message will appear:

```
<Fail @ xxxx
```

Saving a program overwrites the previous one. There is no way to recover it since both occupy the same space.

## AUTORUNNING

To autorun a program:

1. Make sure there is a program in EPROM (from above).

2. Make sure jumper W1 [3-4] and [17-18] are installed.

If you push the reset button, the program should autoexecute. If there are any errors, the program will stop (assuming you have not trapped them with ON ERROR) and display the error message.

## PREVENTING AUTORUN

When troubleshooting a program, it's not always convenient for an autoexecute file to run. This is

especially true if the program has been configured to ignore the < ESC> key.

To prevent autorun, remove jumper W1[17-18].

Later, if you wish to SAVE or LOAD a program, reinstall this jumper. You may do so even if the power is on and a program is running. Remember to discharge any static electricity before installing or removing the jumper.

## LOADING A PROGRAM

There are times when you may wish to temporarily modify or otherwise test out a change to a program. Since the program is loaded into RAM, modifications can be made without affecting the program in EPROM. If you find out that modifications are not desirable or did not work, you can restore the original program to RAM using the LOAD command.

To modify a program before running it on power up, remove jumper W1[17-18] first. Power up or reset the card. Re-install jumper W1[17-18]. Type `load`. You can now list the program and otherwise make modifications.

## SAVING DATA TO EPROM

Additional data, such as strings and constants, can be saved to the EPROM in U3. An external programmer is used to save data to the EPROM. Data cannot be saved to U3 through CAMBASIC.

Data is saved at the top of the EPROM memory. The upper 2K bytes are always available as RAM memory size absolutely limits the program to 30K.

If you need more than 2K bytes of data, you can save data "on top" of the CAMBASIC program in U3. A good way to determine how much memory is available is to perform a PRINT SYS(0) in the immediate mode (program not running). Subtracting 31900 from the number of bytes returned will tell you the approximate number of bytes available for data.

The best way to make sure your data will not write over the program is to perform the following steps:

First, put a remark statement that you can recognize. One is "end of program". Next, save your program to the EPROM using the CAMBASIC SAVE command.

Remove the EPROM and read it from your EPROM programmer. Using your programmer, go into the mode where you can examine data. Look for the remark statement at the end of the program. Instruct the programmer to put your data starting at the next even page boundary (for example 5000H, 5100H, and so on).

## DESCRIPTION

The RPC-150 has two serial ports that can be used for interfacing to a printer, terminal, RS-485 network, or other serial devices. This chapter describes their characteristics and how to use them. Frequent references are made to commands listed in the *CAMBASIC Programming Manual*. Please refer to this manual for more information.

Serial ports are numbered COM1 and COM2. COM1 is used for program development. During run time, it can be used for other functions. COM2 is a general purpose port and can be used as either RS-232 or RS-422/485.

Both ports support XON/XOFF protocol to control data transmission. Each port has a 256 character interrupt driven input and output buffer. This allows characters to be sent out (using PRINT) without slowing down program execution. However, if the PRINT buffer fills, program execution is suspended until the buffer empties. Both ports have a 256 character input buffer. When more than 256 characters have been received, excess ones are ignored.

The baud rate, parity, data length, and stop bit length are changed using the CONFIG BAUD command.

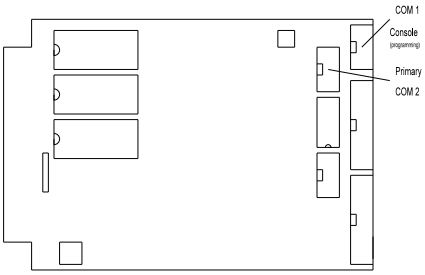



**Figure 4-1 Serial ports**

### COM1 SERIAL PORT

COM1 is J1 and is called the Console port on the card. This port uses a VTC-10 serial cable to connect external serial devices to the port. The cable consists of a 10 pin IDC connector wired to a DB-9 connector. The connector plugs directly into a 9 pin serial port connector on a PC.

This port is normally used for programming. During run time it may be used as a general purpose serial port. When used for programming or with the INPUT statement, it will accept ASCII character values from 0 to 127. When used with the INKEY$ and COM$ functions, it will return ASCII values from 0 to 255.

### COM2 SERIAL PORT

COM2 is an RS-232 port. It also uses a VTC-10 serial cable, described above. COM2 is identical to COM1 except that COM2 has 2 hardware handshaking lines, CTS and RTS. When RTS goes low, the RPC-150 is held off from transmitting out COM2. The status of this port is read by the BIT statement. The example below returns the status of the RTS line:

```
100 B = BIT(130,5)
```

If B = 1, transmission is held off.

The CTS line may be set high or low to hold off communication. Line 400 sets CTS low and 500 sets it high.

```
400 BIT 128,4,1
500 BIT 128,4,0
```

The CONFIG BAUD statement sets the configuration of this port.

## ACCESSING SERIAL BUFFERS

You can access COM1 and COM2 buffers in three ways:

1. INPUT statement. This removes all characters in the buffer up to the terminator character and puts them into a variable.

   When using the INPUT statement, program execution is suspended until a < cr> (Enter key) is received. Whether this is a problem depends on your particular application.

   INPUT strips bit 7 on the COM1 port. This means ASCII characters from 0 to 127 are received. The INPUT statement can return a maximum string length of 150 characters.

2. INKEY$(n) function. Characters are removed one at a time. A null string is returned when the buffer is empty.

In this mode, you have access to the full 256 bytes. If you don't read the buffer and the buffer fills, all subsequent characters are discarded. INKEY$(n) may be used anywhere in the program.

3. COM$(n) retrieves all characters in the buffer, including < cr> 's and other control codes. This function is commonly used with ON COM$ multitasking statement. You can retrieve 128 of the 256 bytes in the serial buffer at one time.

## SERIAL PORT FILE NUMBERS

CAMBASIC references the serial I/O ports by file numbers, similar to DOS. The following table shows the corresponding file number to serial I/O port and how they are used with the various ports.

| Description | File | Examples |
|---|---|---|
| COM1 | 1 | PRINT "Hello"<br>PRINT #1,"Hello"<br>INPUT A$<br>INPUT #1,A$<br>A$ = INKEY$(1) |
| COM2 | 2 | PRINT #2,"Hello"<br>INPUT #2,A$<br>A$ = INKEY$(2) |

COM1 is J1, the console port. COM2 is J4, the primary port.

## COMMANDS

The following is a list of CAMBASIC commands used for serial I/O. Variations for many commands not listed here. These commands and functions are explained in the *CAMBASIC Programming Manual*.

| Command | Function |
|---|---|
| CLEAR COM$ | Clears serial input buffer |
| COM$ | Returns string from buffer |
| CONFIG BAUD | Sets serial port parameters |
| CONFIG COM$ | Configures port for ON COM$(n) interrupt |
| INKEY$ | Returns a character from the serial buffer |
| INPUT | Receives string from port |
| LIST | Outputs program listing |
| ON COM$ | Calls subroutine on serial input |
| PRINT | Outputs data in various formats |
| TAB | Tabs to predetermined positions |

## SERIAL CABLE PIN OUT

The following is the pin out between the IDC connector for the SBS-150 and the DB-9 connector to the PC or terminal.

| IDC | DB-9 | Description |
|---|---|---|
| 1 | 4 | DCD |
| 2 | 3 | RXD |
| 3 | 2 | TXD |
| 4 | 1 | DTR |
| 5 | 5 | Ground |
| 6 | n/c | DSR |
| 7 | 8 | CTS out |
| 8 | 7 | RTS in* |
| 9 | n/c | + 5 V |
| 10 | n/c | RI |

J1 and J4 connector pin out.

| Pin | J1 | J4 | Direction |
|---|---|---|---|
| 1 | n/c | n/c | |
| 2 | RXD | RXD | Input |
| 3 | TXD | TXD | Output |
| 4 | n/c | n/c | |
| 5 | Gnd | Gnd | |
| 6 | n/c | n/c | |
| 7 | n/c | CTS | Output |
| 8 | n/c | RTS | Input |
| 9 | + 5V | + 5V | source |
| 10 | n/c | n/c | |

## INTRODUCTION

The RPC-150 is available with 32 of RAM. RAM is in socket U1. RAM may be battery backed by installing a DS-1213C in socket U1. RAM is installed on top of U1.

Battery life depends upon RAM power consumption, temperature, and amount of time the board is operating. Generally, a battery life from 5 to 10 years can be expected.

This chapter discusses installing a battery backup for RAM, saving and retrieving variables, and running assembly language programs. Figure 5-1 shows the location of U1.

If program and data are battery backed, the UNNEW command may be used to restore the program. Variables used by the Basic program are cleared, however. Data POKEd into RAM is saved.

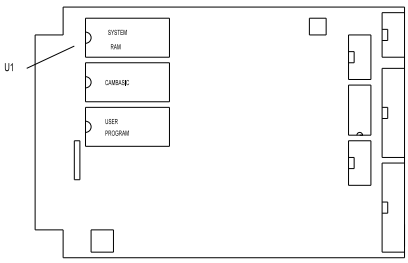



**Figure 5-1 Data memory**

## BATTERY BACKUP

A Dallas Semiconductor DS-1213C is used to battery backup RAM when power is off. Battery life depends upon RAM size, type, and time the RPC-150 has power applied to it. You can expect the battery to last between 5 to 10 years at 25°C. Battery life decreases by 1/2 at 50°C.

To install a DS-1213C, remove the RAM chip in U1, install the DS-1213C, and install the RAM chip on top of the module.

## STORING VARIABLES IN RAM

The term "variables" in this context includes numbers, strings, arrays, recipes, and formulas as applied to your application.

Programs, CAMBASIC variables, and variables you POKE and PEEK to are from 8000H to FFFFH. The program and basic number variables (A, B(15), ect.) are 'on top' of the program and are cleared on reset. String variables (e.g. c$) are below the stack. The stack requires 255 bytes and string space will depend upon how much you CLEARed on power up. If you do not use the CLEAR statement, assume 500 bytes are available for strings. Data you peek and poke to must be between the end of number variables and start of strings.

The procedure to determine the 'safe' area to Poke variables follows. First, download and run your program. What you want to do is force the basic to allocate space for most all of the variables in your program. Next, exit your program and perform a PRINT SYS(2), SYS(1) in the immediate mode. This will print out the address for the bottom of the stack and top of the program with variables. Subtract the number used in your CLEAR statement from the first number printed. If you did not use the CLEAR statement, subtract 500. In case your program did not execute completely, you may want to add 500 to the second number. This range is where you can POKE to.

Many times you will start with a very small program and add to it. In this case the above procedure is not adequate. The best way to handle this situation is to figure out the number of bytes used for string variables in the program. This determined by the CLEAR statement. If you do not use the CLEAR statement, then you can figure the bottom of the stack and string space start at address 0FD00H. When you start using POKE statements, assign new variables BELOW this address. To put it another way, count down and not up as you add POKE locations. Remember when using string and floating point POKEs the address specified is the starting address and will go up.

PEEK and POKE commands store and retrieve values from memory. For example:

```
20 POKE 60000, A
```

puts the value of A into address 60000.

Use the PEEK statement to retrieve the variable:

```
50 B = PEEK(60000)
```

You can store and retrieve arrays, strings, and variables in this way. There are many variations of PEEK and POKE statements. A list of CAMBASIC commands appears at the end of this chapter.

## ASSEMBLY LANGUAGE INTERFACE

Use the CAMBASIC CALL statement to execute an assembly or C language program.

A specific area of RAM should be reserved for the program. This is to prevent strings and variables from corrupting that area of RAM. Use the SYS(1) and SYS(2) statements to do this. SYS(1) returns the low memory location while SYS(2)-500 returns the upper location. Run the program first to make sure variable memory has been allocated before running these SYS commands. Failure to do so may result in address returned that are not really free for assembly language programs.

There are several ways to put a program in memory, depending upon your application.

1. Use DATA statements and POKE the code into segment 0 RAM.

2. Write a program to download code. Some applications are connected to a larger system which "initializes" its systems. Using INKEY$ or COM$, code is received and then poked into memory using POKE$.

3. Read the code from the EPROM (U3) (using INP) and transfer it to RAM (using POKE). You would have to use an external programmer to place the code above CAMBASIC code.

In all cases, it is best to load code into RAM from a "secure" source. Even though RAM is battery backed, over time there is the possibility it could be corrupted.

Below is an example of loading and running an assembly language program.

```
100  FOR N = &FB00 TO &FB0C
110  READ A
120  POKE N,A
130  NEXT

900  DATA &DB, 2, &47, &E6, &FE, &D3
910  DATA 2, &78, &F6, 1, &D3, 2, &C9

2000 CALL &FB00
```

Lines 100 to 130 load the program into RAM.

Line 2000 calls the program. It toggles J2 line 13.

## COMMANDS

The following is a list of CAMBASIC commands used with RAM.

| Command | Function |
|---|---|
| CALL | Calls an assembly language routine |
| CLEAR | Clears strings and allocates string space |
| PEEK | Returns a byte |
| DPEEK | Returns a 16 bit value |
| PEEK$ | Returns a string |
| FPEEK | Returns a floating point number |
| POKE | Stores a byte |
| DPOKE | Stores a 16 bit value |
| POKE$ | Stores a string |
| FPOKE | Stores a floating point number |

## INTRODUCTION

Digital I/O lines are used to interface with opto-module racks, switches, low current LED's, and other TTL devices. The RPC-150 has 47 of these lines available through J2 and J3. Seven of these lines are high current outputs, capable of sinking 75 to 200 ma. Eight lines on J2 are shared by the keypad connector, J5. When the keypad is used, 8 of the 47 lines are not available.

Eight, 16, or 24 position opto racks are connected to J2 or J3. These opto racks accept G4 series opto modules. G4 series opto modules are used to sense the presence of AC or DC voltages or switch them. Maximum switching current is 3 amperes.

### *WARNING:*

Apply power to the RPC-150 before applying a voltage to the digital I/O lines to prevent current from flowing in and damaging devices. If you cannot apply power to the RPC-150 first, contact technical support for suggestions appropriate to your application.

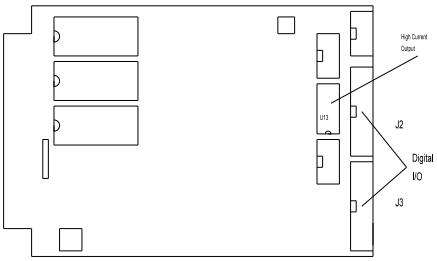



**Figure 6-1 Digital I/O**

## DIGITAL I/O PORT

Digital I/O lines on the RPC-150 are supplied by an 82C55 chip. The chip's lines go to connectors J2 and J3.

The lines on J2 and J3 are divided into 3 eight bit groups. Ports A and B can be configured as all inputs or outputs. Port C can be programmed as one group of 8 inputs or outputs or as two groups of four lines (upper and lower C). The four lines in upper and lower C can each be programmed as all inputs or outputs. Port B, bit 7 on J2 must be used as an output.

When a line is configured as an output, it can sink a maximum of 2.5 ma at 0.4V and can source over 2.5 ma at 2.4V. Outputs sink 15 ma at 1.0V.

J2 and J3 are accessed using CAMBASIC LINE, OPTO, INP, and OUT statements. LINE reads or writes to a port based on the connector pin number. OPTO reads or writes to an opto module based on its position in an MPS opto rack. INP and OUT access a byte of data at a port.

The base address for J2 is 512 and J3 is 0 when using CONFIG PIO, INP, and OUT statements. CONFIG PIO statement is used to configure the 8255 lines. Upon reset, lines are configured for inputs. Use CONFIG PIO to configure lines for outputs and inputs.

J2 and J3 are accessed using LINE or OPTO statements according to the table below.

| Connector No | LINE # terminal | OPTO rack position |
|---|---|---|
| J3 | 1 - 25 | 0 - 23 |
| J2 | 101 - 124 | 100 - 122 |

J2 port B is connected to a high current sink through U13. See "*High current output*" later in this chapter. Line 107 is not available.

J2 port C is shared with the keypad port J5. If you are using a keypad through J5, these 8 lines are not available.

**Pull up resistors**

Digital I/O lines at J2 and J3 are pulled up to + 5 volts through a 10K resistor pack.

These pull ups makes interfacing to switches and "open collector" TTL devices easy. See "Interfacing to Switches and other devices" below.

**High current output**

Eight lines at J2 can be used as high current drivers. These outputs will switch loads to ground. Outputs are controlled by Port B on the 82C55. Its address is 513. Port B bits 0-6 are used to control the high current port.

Logic outputs from this port are inverted. That is, when a 1 is written to the high current port, the output is switched on and goes low.

The output driver chip, U13, can be replaced with a DIP shunt jumper so it is like the other lines at J2. Install the jumper so U13 pin 1 goes to pin 18.

***NOTE:*** Outputs at the high current lines are not compatible with TTL logic levels and should not be used to drive other logic devices.

Each of the high current outputs can sink 100 ma at 50V.

## *WARNING:*

External supplies using the high current outputs must be tied to J2, pin 26 and NOT the power connector. Failure to do so can produce a ground loop and cause erratic operation.

The thermal time constant of the package is very short, so the number of outputs that are on at any one time should include those that overlap even for a few milliseconds.

Incandescent lamps have a "cold" current of 11 times its operating current. Lamps requiring more than 50 ma should not be used.

Protection diodes must be used with inductive loads. Refer to figure 6-2

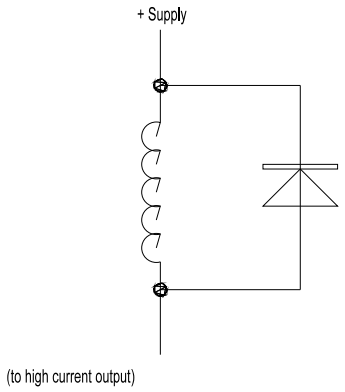



**Figure 6-2 Inductive load protection**

Do not parallel outputs for higher drive. This could result in damage since outputs will not share current equally.

**Interfacing to an opto-module rack**

J2 and J3 I/O lines can be interfaced to an MPS-8, 16, or 24 position opto module rack. Lines not going to an opto module connect to a screw terminal on the MPS-XX series boards. This feature allows you to connect switches or other TTL type devices to the digital I/O lines. The MPS-XX series boards accept G4 series modules.

Use the OPTO command to access and control G4 opto modules. The LINE command is used to access individual lines on the STB-26 or MPS-XX board.

A CMA-26 connects J2 and J3 on the RPC-150 to the MPS-XX board. Cable length should be less than 2 feet for the 8 position rack and 18 inches for the 16 and 24 positions. Excessive cable lengths cause a high voltage drop and consequently unreliable operation. Connect + 5V and ground to the opto racks.

You must configure the 8255 ports for outputs using the CONFIG PIO statement. Use the following table to determine the corresponding opto channel for a particular 82C55 port:

| Opto channels | 82C55 port | Connector | Addr. |
|---|---|---|---|
| M100-M103 | Lower C | J2 | 514 |
| M104-M107 | Upper C | J2 | 514 |
| M108-M115 | A | J2 | 512 |
| M116-M122 | B | J2 | 513 |
| M123 | not available | | |
| M0-M3 | Lower C | J3 | 2 |
| M4-M7 | Upper C | J3 | 2 |
| M8-M15 | B | J3 | 1 |
| M16-M23 | A | J3 | 0 |

"Opto channel" is the position as marked on the MPS-xx board. The channel number is preceded by a 'M' character on the MPS board. When connecting J2 to an opto rack, add 100 to the number on the rack. J2 has a high current output on port B (channels M8-M15). Replace U13 with a shunt jumper to operate normally.

You will have to group your inputs and outputs on the rack to make best use of the space. See *Digital I/O Programming Example* later in this chapter for information about using these commands. Pay attention to how LINE differs from OPTO in addressing and polarity.

To turn on an opto module, an output line must be low. A module is turned off by writing a '1' to a channel.

The logic at J2 port B, with the high current outputs installed is just the reverse. A '1' at a line causes the module to turn ON. The OPTO command sets the output to the proper polarity when U13 is replaced with a dip shunt jumper.

High current outputs at J2 port B are optionally configurable as TTL I/O by replacing U13 with a DIP shunt jumper. This will keep the logic compatible with ports A and C. If opto channels 0-6 are used as inputs, then U13 must be replaced by a DIP shunt jumper.

**Interfacing to switches and other devices**

The STB-26 terminal board provides a convenient way of interfacing switches or other digital I/O devices. Lines at J2 and J3 are connected to the STB-26 with a CMA-26 ribbon cable. Digital devices are then connected to the screw terminals on the STB-26.

Switches may be connected directly to a line. 10K pull-up resistors on board set all inputs high, or 1. A switch closure to ground at a line is read as a 0.

**Configuring digital I/O lines**

Lines are configured during program execution using the CONFIG PIO command. On power up or reset, all lines are inputs.

When a line is configured as an output, it can sink a maximum of 2.5 mA at 0.4V and can source a minimum of 2.5 ma at 2.4V. When driving opto modules, the outputs sink 15 mA at 1.0V.

**Digital I/O programming example**

The following example reads a switch at port A, bit 3 (J2-25), reads an opto module at channel 1 and writes an opto module at channel 5. A LED is controlled at J2-10 (port B, bit 0).

```
200 D = BIT(0,3)     :'Read switch status, port A
210 F = OPTO(101)    :'read opto module, ch. 101
220 OPTO 3,ON        :'write module, channel 3
230 BIT 1,0,0        :'turn on led at J2-10
240 BIT 1,0,1        :'turn off led at J2-10
250 A = LINE(103)    :'Reads pin 3 at J2
260 LINE 114,1       ;'Set line 114 high
```

**Connector pin out - J2**

| Pin # | 82C55 Port | Description | Opto Channel |
|---|---|---|---|
| 19 | Port A, line 0 | | 108 |
| 21 | Port A, line 1 | | 109 |
| 23 | Port A, line 2 | | 110 |
| 25 | Port A, line 3 | | 111 |
| 24 | Port A, line 4 | | 112 |
| 22 | Port A, line 5 | | 113 |
| 20 | Port A, line 6 | | 114 |
| 18 | Port A, line 7 | | 115 |
| 10 | Port B, line 0 | High current | 100 |
| 8 | Port B, line 1 | High current | 101 |
| 4 | Port B, line 2 | High current | 102 |
| 6 | Port B, line 3 | High current | 103 |
| 1 | Port B, line 4 | High current | 104 |
| 3 | Port B, line 5 | High current | 105 |
| 5 | Port B, line 6 | High current | 106 |
| 7 | Port B, line 7 | not available | |
| 13 | Port C, line 0 | Shared with J5 | 116 |
| 16 | Port C, line 1 | Shared with J5 | 117 |
| 15 | Port C, line 2 | Shared with J5 | 118 |
| 17 | Port C, line 3 | Shared with J5 | 119 |
| 14 | Port C, line 4 | Shared with J5 | 120 |
| 11 | Port C, line 5 | Shared with J5 | 121 |
| 12 | Port C, line 6 | Shared with J5 | 122 |
| 9 | Port C, line 7 | Shared with J5 | 123 |
| 26 | | | Ground |
| 2 | | | + 5V |

Opto Channel is computed by adding 100 to the number on the MPS rack. The position number is preceded by the letter 'M' on the rack.

**Connector pin out - J3**

| Pin # | 82C55 Port | Description | Opto Channel |
|---|---|---|---|
| 10 | Port A, line 0 | | 0 |
| 8 | Port A, line 1 | | 1 |
| 4 | Port A, line 2 | | 2 |
| 6 | Port A, line 3 | | 3 |
| 1 | Port A, line 4 | | 4 |
| 3 | Port A, line 5 | | 5 |
| 5 | Port A, line 6 | | 6 |
| 7 | Port A, line 7 | | 7 |
| 19 | Port B, line 0 | | 8 |
| 21 | Port B, line 1 | | 9 |
| 23 | Port B, line 2 | | 10 |
| 25 | Port B, line 3 | | 11 |
| 24 | Port B, line 4 | | 12 |
| 22 | Port B, line 5 | | 13 |
| 20 | Port B, line 6 | | 14 |
| 18 | Port B, line 7 | | 15 |
| 13 | Port C, line 0 | | 16 |
| 16 | Port C, line 1 | | 17 |
| 15 | Port C, line 2 | | 18 |
| 17 | Port C, line 3 | | 19 |
| 14 | Port C, line 4 | | 20 |
| 11 | Port C, line 5 | | 21 |
| 12 | Port C, line 6 | | 22 |
| 9 | Port C, line 7 | | 23 |
| 26 | | | Ground |
| 2 | | | + 5V |

Opto Channel numbers correspond to the number preceded by the letter 'M' on the opto rack.

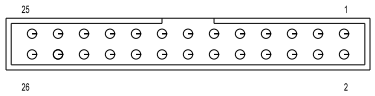



**Figure 6-3 Digital I/O connector pin out (viewed from top)**

## COMMANDS

The following tables shows the CAMBASIC commands used for digital I/O.

| Command | Function |
|---|---|
| BIT | Function returns status of bit at an I/O address |
| BIT | Command sets a bit at an I/O address |
| CONFIG PIO | Configures J3 I/O port |
| INP | Returns a byte from an I/O address |
| LINE | Returns status of a line |
| LINE | Sets line high or low |
| OPTO | Sets an opto module output |
| OPTO | Reads opto module status |
| OUT | Writes a byte to an I/O address |
| PULSE | Reads or writes a pulse at a port. |

See also ON BIT, ON COUNT, ON INP, and related statements.

## DESCRIPTION

The RPC-150 has a battery backed Calendar/clock option using the DS-1216EM. When used in conjunction with the DATE$ and TIME$ commands, the current date and time can be set and read. The DS-1216EM is accurate to 1 minute/month at 25°C. Battery life is a minimum of 5 years at 25°C and 2 years when at 50°C.

## INSTALLATION

The real time clock is installed in socket U3. Remove the IC in U3. Install the DS-1216EM in U3. Install the IC previously removed into the top of the DS-1216EM.

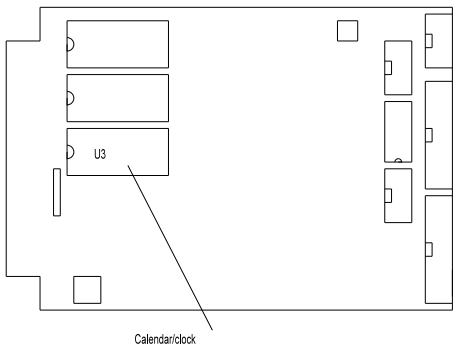



**Figure 7-1 Calendar/clock installation**

## SETTING DATE AND TIME

The clock must be turned on before it is used. This need be done only once. To turn on the clock, type:

```
CONFIG CLOCK ON
```

The date and time can be set while running a program or in the immediate mode. Date and time are treated as strings and not numbers. To set the date and time:

```
date$="07-25-94"
time$="15:03:00"
```

To retrieve date and time as part of a program:

```
2000 DA$ = DATE$(0)
2010 TI$ = TIME$(0)
```

You can also print the date and time in the immediate mode:

```
pr time$(0)
15:04:03
```

The clock is turned on and off using the CONFIG CLOCK statement.

## COMMANDS

The following is a list of CAMBASIC commands for the calendar/clock.

| Command | Function |
|---|---|
| CONFIG CLOCK | Configures clock |
| DATE$ | Sets date |
| DATE$(0) | Returns date |
| TIME$ | Sets time |
| TIME$(0) | Returns time |

## INTRODUCTION

16 position keypads are plugged into keypad port J5. Keys are arranged in a 4 x 4 matrix format. A key is recognized when a row and a column connect.

CAMBASIC automatically scans and debounces the keypad every debounce time. Debounce time is fixed at 80 ms. Keypad presses may be returned either as a number from 1 to 16 or as an ASCII character. The ASCII character returned corresponds to those on Remote Processing's KP-1 keypad. Character assignments are changed using the SYS(8) function.

Keypads from Remote Processing simply plug into J5. Keypad cable length should be limited to less than 5 feet.

If the keypad port is not used, it may be used as a general purpose digital I/O port.

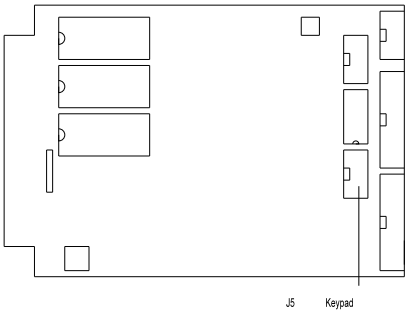



**Figure 9-1 Keypad connector**

## PROGRAMMING EXAMPLE

The following example sets up CAMBASIC to scan a 16 position keypad. The results are echo'ed to the display and the speaker is sounded when a key is pressed.

```
10 CONFIG PIO 0,0,1,1,0,512
20 'Optionally change keypad character 'B'
30 ' to the letter 'M'
40 POKE SYS(8)+7,77
60 ON KEYPAD$ GOSUB 500
70 PRINT " Enter a number";
100 'loop for this example
110 GOTO 100

500 A$ = KEYPAD$(0)
510 IF A$ = "C" THEN ..clear_beep
520 IF A$ = CHR$(13) THEN ..enter
530 PRINT A$;
540 B$ = B$+A$
560 RETURN

600 ..clear_beep
610 B$=""
630 DELAY .4
650 PRINT CHR$(12); "        ";CHR$(12);
660 RETURN

700 ..enter
710 FL = 1
730 RETURN
```

**Program explanation**

Lines 10-80 set up the parameters for the keypad. Lines 500 to 730 process the key press. If a "C" or "#" is pressed, it is an exception and is handled that way. Otherwise, the character is displayed and stored.

Lines 700 to 730 process the "enter" key ('#'). The enter flag, FL, is set to a 1 to indicate to another part of the program that B$ has complete data.

The KEYPAD$(0) function returns a single character string that has been assigned to a particular key. Characters are assigned using the SYS(8) statement.

## KEYPAD PORT PIN OUT - J5

The keypad port uses port C from an 82C55. Lower port C is an input. Upper port C is output.

The table below lists J5's pin out, 82C55 port and bit, and its intended function.

| Pin | 82C55 Port/bit | Function |
|---|---|---|
| 1 | C/0 | Row 1 |
| 2 | C/6 | Column 3 |
| 3 | C/5 | Column 2 |
| 4 | C/1 | Row 2 |
| 5 | C/2 | Row 3 |
| 6 | C/4 | Column 1 |
| 7 | C/7 | Column 4 |
| 8 | C/3 | Row 4 |
| 9 | nc | |
| 10 | | Ground |

Ground is not needed for keypad operation.

## COMMANDS

The following is a list of CAMBASIC commands for the keypad.

| Command | Function |
|---|---|
| INPUT KEYPAD$ | Input data from a keypad |
| KEYPAD$(n) | Returns last key from keypad port |
| ON KEYPAD$ | Causes a program branch when a key is pressed |
| SYS(8) | Returns keypad string address |

## DESCRIPTION

Pin 16 on the card edge connector is the speaker output from the 64180 CPU chip. This port can be used to drive a speaker. The SOUND command is used to generate a frequency.

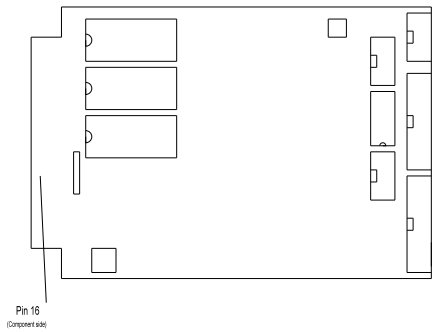



**Figure 9-1 Speaker pin location**

**Connecting a speaker**

Refer to figure 9-2 below for circuit connections to a speaker. The series resistor determines the volume. the Capacitor sets the lower frequency limit. Generally, values from 100 uF to 470 uF are adequate. The speaker may be any value but those with 50 ohms or greater produce higher sound output.

***WARNING:*** Do not connect pin 16 directly to a speaker, ground, or + 5V, even momentarily, as damage to the CPU will result.

## SYNTAX

SOUND *frequency[,time]*

Where: *frequency* = 20 to 15000 Hz
*time* = time in seconds

SOUND stops executing when CAMBASIC is not running a program. The output will be at *frequency* until SOUND is executed without a parameter or the optional *time* is timed out.

*frequency* is not exact. It is, however, accurate enough for most alarm or audio feedback applications.

***NOTE:*** When SOUND is used with a *time* parameter, program execution is suspended until it is timed out.

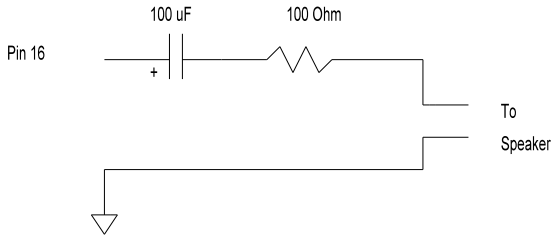



**Figure 9-2 Interface schematic**

**Programming example**

The following example produces tones from 200 to 5000 hz and back down again in 500 hz increments.

```
10 FOR N = 200 TO 5000 STEP 500
20  SOUND N,.5
30 NEXT
40 FOR N = 5000 TO 200 STEP -500
50  SOUND N,.5
60 NEXT
70 GOTO 10
```

To stop program execution, press the < esc> key.

# TECHNICAL INFORMATION

## ELECTRICAL

**CPU**
64180, 9.216 Mhz clock

**Memory**
CAMBASIC, 32K ROM

Programming and data is 32K RAM

Program is 32K EPROM (U3)

Memory speeds are 80 ns or faster

**Digital I/O**
The RPC-150 has 47 digital I/O lines. 23 are from J2. The other 24 are from J3. J2 has 7 high current outputs, which may be jumpered for inputs. The keypad port, J5, uses 8 of the 23 lines on J2. All ports use an 82C55 for interfacing.

The specifications below are for all digital I/O except for the eight high current lines at J2.

| | |
|---|---|
| Drive current | 2.5 ma maximum per line, sink or source. TTL compatible. |
| Output low voltage | 0.45V max at 2.5 mA, 1V max at 15 mA for opto rack. |
| Output high volts | 2.4V minimum, sink or source at rated current. |

All digital input lines are TTL compatible.

**High current output at J2**

7 of the 24 lines can drive up to 500 ma at 50V. Refer to *CHAPTER 6, DIGITAL AND OPTO PORTS* for limitations.

**Keypad input**
8 lines accept a 16 position matrix keypad. Scanning and debounce performed in CAMBASIC.

**Serial ports**
Two RS-232D serial ports. All have RxD and TxD lines. COM1 has only these lines. COM2 also has CTS and RTS line. Baud rates from 600 to 38.4K, 7 or 8 data bits, parity even, odd, or none, 1 or 2 stop bits.

**EPROM and programmer**
Accepts 29C256 or equivalent EPROM.
Size:32K
Speed:200ns or faster.

**Power requirements**
+ 5VDC ±5% at 100 ma.
RS-232 voltages generated on card.
Current consumption does not include any opto-modules or other accessories.

## MECHANICAL

**Size:** 4.5" x 6.0"
Maximum height: 0.675", no cables installed.

Mounting holes: 4 each corner. Hole size is 0.156" dia. See drawing.

## MEMORY AND I/O MAP

**Memory**

| Description | Address |
|---|---|
| CAMBASIC, U2 | &00000 - &07FFF |
| RAM, U1 | &08000 - &0FFFF |

I/O

| | |
|---|---|
| J3 Digital | &0000 - &0003 |
| J2 Digital | &0200 - &0203 |
| Keypad | &0202 - &0202 |
| Internal processor | &0080 - &00BF |
| Program EPROM(U3) | &0300 - &7FFF |

Only I/O address &8000 to &FFFF are available off card. No memory addresses are available off card.

## JUMPER DESCRIPTIONS

A * after a jumper position indicates factory default and is jumpered.

| Jumper | Description |
|---|---|
| W1 | U3 control |
| [3-4]* | U3 write enable |
| [17-18]* | U3 chip select |

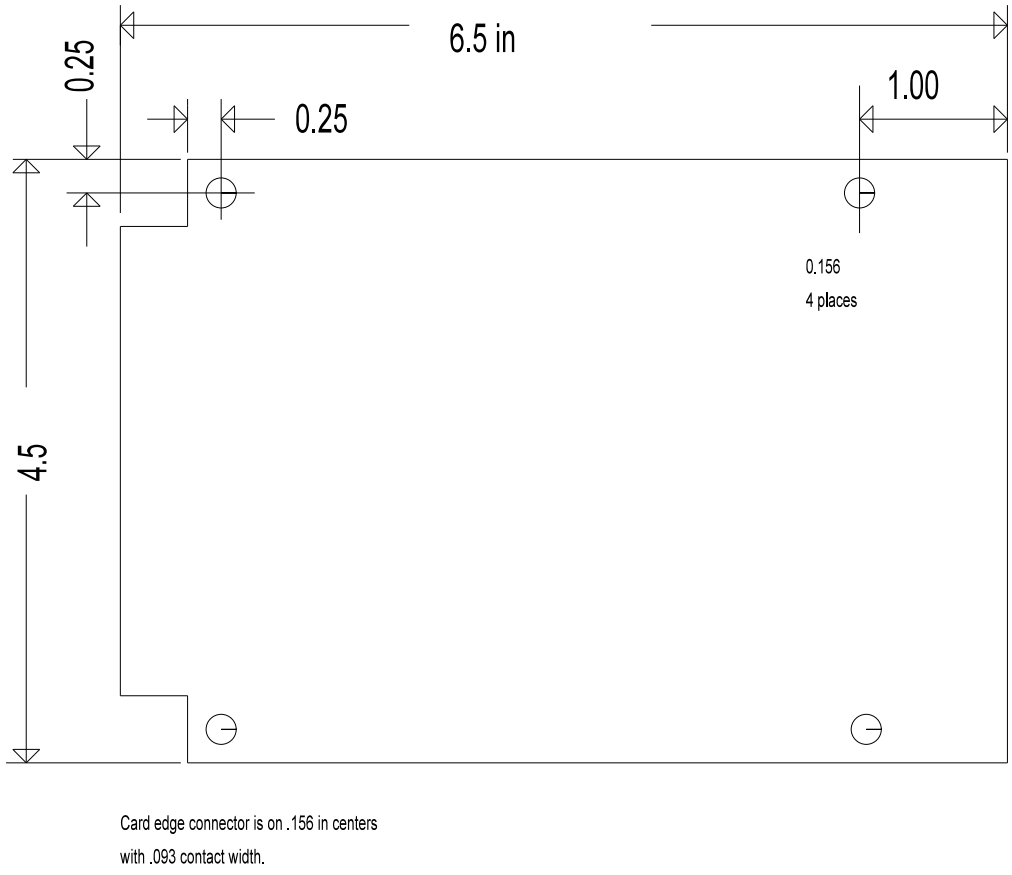
* = default

**Edge connector**

| Line | Signal | Description |
|---|---|---|
| A | + 5 | Power |
| B | D0 | Data 0 |
| C | D2 | Data 2 |
| D | D4 | Data 4 |
| E | D6 | Data 6 |
| F | A0 | Address A0 |
| H | A2 | Address A2 |
| J | A4 | Address A4 |
| K | A6 | Address A6 |
| L | A8 | Address A8 |
| M | A10 | Address A10 |
| N | A12 | Address A12 |
| P | A14 | Address A14 |
| R | IWR* | I/O write |
| S | HOLD* | CPU hold |
| T | - | not used |
| U | INT0 | Interrupt 0 |
| V | CLK | CPU clock |
| W | PS* | Peripheral select |
| X | + 12.7 | Power |
| Y | - | not used |
| Z | Gnd | Power ground |
| | | |
| 1 | + 5V | Power |
| 2 | D1 | Data D1 |
| 3 | D3 | Data D3 |
| 4 | D5 | Data D5 |
| 5 | D7 | Data D7 |
| 6 | A1 | Address A1 |
| 7 | A3 | Address A3 |
| 8 | A5 | Address A5 |
| 9 | A7 | Address A7 |
| 10 | A9 | Address A9 |
| 11 | A11 | Address A11 |
| 12 | A13 | Address A13 |
| 13 | A15 | Address A15 |
| 14 | IRD* | I/O Read strobe |
| 15 | - | not used |
| 16 | Sound | Speaker port |
| 17 | - | not used |
| 18 | RES* | Reset input |
| 19 | INT1 | Interrupt input |
| 20 | - | not used |
| 21 | -12V | Power |
| 22 | Ground | Power ground |

**Serial Port pin out**

| Pin # | J1 Signal | J4 Signal |
|---|---|---|
| 1 | NC | NC |
| 2 | RxD | RxD |
| 3 | TxD | TxD |
| 4 | NC | NC |
| 5 | Gnd | Gnd |
| 6 | NC | NC |
| 7 | NC | CTS |
| 8 | NC | RTS |
| 9 | + 5 | + 5 |
| 10 | NC | NC |




**RPC-150 board outline**