

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 applications where the failure or malfunction of a component may directly threaten life or injury. It is a Condition of Sale that the user of Remote Processing products in life support applications assumes all the risk of such use and indemnifies Remote Processing against all damages.

P/N 1043  
Revision: 1.0

# TABLE OF CONTENTS

---

DESCRIPTION . . . . .	1	SETTING DATE AND TIME . . . . .	23
MANUAL ORGANIZATION . . . . .	2	COMMANDS . . . . .	23
MANUAL CONVENTIONS . . . . .	2	INTRODUCTION . . . . .	24
Symbols and Terminology . . . . .	2	CONNECTING DISPLAYS . . . . .	24
TECHNICAL SUPPORT . . . . .	3	WRITING TO THE DISPLAY . . . . .	24
INTRODUCTION . . . . .	4	PROGRAMMING EXAMPLE . . . . .	24
OPERATING PRECAUTIONS . . . . .	4	DISPLAY TYPES . . . . .	24
EQUIPMENT . . . . .	4	DISPLAY CONNECTOR PINOUT - J4 . . . . .	25
FIRST TIME OPERATION . . . . .	5	COMMANDS . . . . .	25
UPLOADING AND DOWNLOADING		INTRODUCTION . . . . .	26
PROGRAMS . . . . .	5	PROGRAMMING EXAMPLE . . . . .	26
Uploading programs . . . . .	5	KEYPAD PORT PINOUT - J5 . . . . .	26
Downloading programs . . . . .	6	COMMANDS . . . . .	27
Other communications software . . . . .	6	DESCRIPTION . . . . .	28
Editing programs and programming hints . . . . .	6	DESCRIPTION . . . . .	29
WHERE TO GO FROM HERE . . . . .	7	PROGRAM EXAMPLES . . . . .	29
TROUBLESHOOTING . . . . .	8	EXTERNAL RESET . . . . .	29
INTRODUCTION . . . . .	9	DESCRIPTION . . . . .	30
SAVING A PROGRAM . . . . .	9	DESCRIPTION . . . . .	31
AUTORUNNING . . . . .	9	PROGRAM EXAMPLE . . . . .	31
PREVENTING AUTORUN . . . . .	10	ELECTRICAL . . . . .	32
LOADING AN EEPROM PROGRAM . . . . .	10	MECHANICAL . . . . .	32
DESCRIPTION . . . . .	11	MEMORY AND I/O MAP . . . . .	33
COM1 SERIAL PORT . . . . .	11	JUMPER DESCRIPTIONS . . . . .	33
COM2 SERIAL PORT . . . . .	11		
Termination network . . . . .	11		
MULTIDROP NETWORK . . . . .	12		
ACCESSING SERIAL BUFFERS . . . . .	14		
SERIAL PORT FILE NUMBERS . . . . .	14		
COMMANDS . . . . .	14		
INTRODUCTION . . . . .	15		
CHANGING MEMORY . . . . .	15		
BATTERY BACKUP . . . . .	15		
STORING VARIABLES IN RAM . . . . .	16		
CORRUPTED VARIABLES . . . . .	16		
ASSEMBLY LANGUAGE INTERFACE . . . . .	17		
COMMANDS . . . . .	17		
INTRODUCTION . . . . .	18		
ON-CARD OPTO RACK . . . . .	18		
Description . . . . .	18		
Installation . . . . .	18		
G5 operation . . . . .	18		
Converting analog measurements . . . . .	19		
DIGITAL I/O PORT . . . . .	19		
Pull up / down resistors . . . . .	19		
High current output . . . . .	20		
Interfacing to an opto-module rack . . . . .	20		
Interfacing to switches and other devices . . . . .	21		
Configuring digital I/O lines . . . . .	21		
Digital I/O programming . . . . .	21		
Connector pinout - J3 . . . . .	21		
COMMANDS . . . . .	22		
DESCRIPTION . . . . .	23		

**DESCRIPTION**

The RPC-30 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 EEPROM programmer saves programs to 30K.

Four position opto rack accepts G4 and G5 series modules. The G4 series is the industry standard digital I/O module. G5 modules are optically isolated analog I/O.

LCD and keypad ports for operator interface. A built in speaker can produce alarm or annunciation tones.

Two RS-232 serial ports, one of which can be configured for RS-422/485.

Watchdog timer resets the card if the program "crashes". The timer is enabled and disabled by software.

24 general purpose digital I/O lines, 8 of which are high current outputs. These lines can connect to another opto rack.

Calendar/clock is battery backed and keeps track of date and time even when power is off.

32K, 128K, or 512K RAM is battery backed to save process variables and other data when power is off.

A serial EEPROM saves program constants in a secure media.

The RPC-30 uses a Z180 CPU operating at 18 Mhz. It can operate stand alone or on a network using the RS-485 port. Its 4.5" x 7" size makes it easy to mount in a NEMA box. Compactness is enhanced by the 4 on-board opto module slots.

CAMBASIC programming language is standard. This language was adapted for the RPC-30 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-30. 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 operate the RPC-30. Using this manual you will be able to:

Interface the RPC-30 to your IBM compatible PC or terminal.

Understand the operation of the RPC-30 hardware using CAMBASIC programming software.

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-30 uses a Zilog Z180 processor. Additional information can be obtained from Zilog, (408-370-8016), or your distributor.

**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.

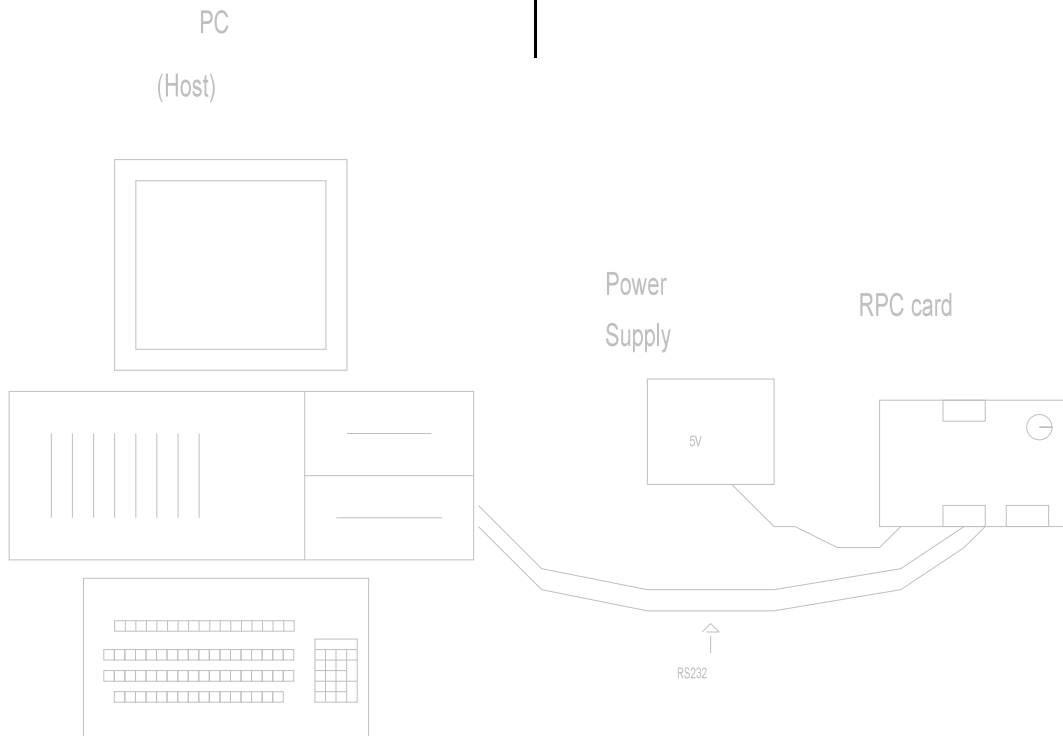
**TECHNICAL SUPPORT**

If you have a question about the RPC-30 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-30* and *CAMBASIC manuals* ready. Sometimes it is helpful to know what the RPC-30 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-30 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 what chapters to read in order to use the various capabilities of the RPC-30. Finally, a troubleshooting section helps out on the most common problems.

**OPERATING PRECAUTIONS**

The RPC-30 is an industrially rated product, 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-30.

1. Ground yourself before handling the RPC-30 or plugging in cables. Static electricity

can easily arc through cables and to the card. Simply touching your PC can greatly reduce the amount of static.

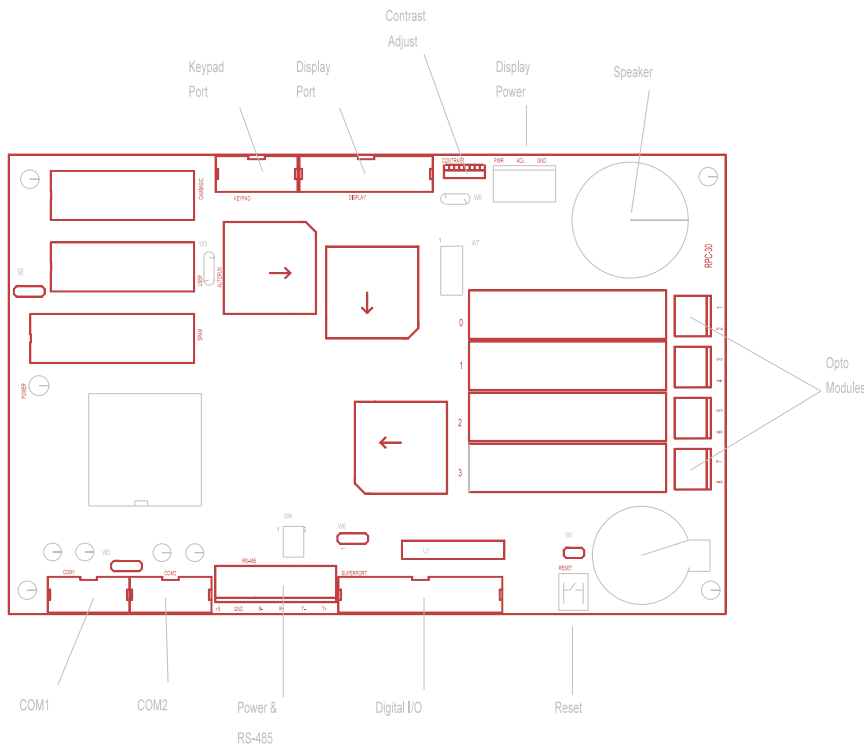
2. Do not insert or remove components when power is applied. While the card is a + 5 volt only system, other voltages generated on the card. Applying them in the wrong sequence can destroy a component.

**EQUIPMENT**

You will need the following equipment to begin using the RPC-30:

- RPC-30 embedded controller
- PC with a serial port and communications program (such as PC SmartLINK)
- or
- Terminal
- VTC-9F serial cable
- + 5, 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-30 Connector Layout**

## FIRST TIME OPERATION

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

RPC-30 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. See

### Technical Information.

1. The RPC-30 needs + 5  $\pm$ 0.25 volts at 100 ma. Any well regulated supply that supplies this will work. Be careful when using "switching" power supplies. Some of these supplies do not regulate properly unless they are adequately loaded. Don't forget that power requirements will increase when opto modules are installed. The G5 series requires about 130 - 150 ma per module.

Make sure power is off. Connect the power supply to the appropriately marked terminals on the RPC-30.

2. You can use either a PC or CRT terminal to program the RPC-30. Connect one end of the VTC-9F connector to then 10 pin COM1 port on the RPC-30. Refer to Figure 2-1 for connector location.

### Using a PC

Connect the VTC-9F serial cable to the PC's COM1 or COM2 port. You may need a 9 pin male to 25 pin female adapter. The VTC-9F 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 male adapter to connect the VTC-9F.

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

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

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-30.

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 at the end of this section.

### 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 you saved it under.
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. 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 will not be a problem.

COM1 on the RPC-30 does not recognize the CTS or RTS lines. The CTS line is pulled high on the RPC-30. The effect of not recognizing these lines is your PC or terminal cannot hold off the RPC-30's transmission. Converse, the RPC-30 cannot hold off the host from sending it data.

**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 form at.

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 VAT 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-30 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.

**NOTE:** Some versions of PC SmartLINK may optionally strip comments before downloading. Check your manual to see if this option is available.

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. This 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 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 information about increasing program speed and editing options.

Instead of uploading and downloading programs, you can save them to the on card EEPROM. This is useful if you are using a terminal to write programs. Make sure the 'AUTORUN' jumper is installed (See WRITING AND 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-30, 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
Run a program at power up or reset (autorun)	3
Know more about serial ports	4
Install a different RAM memory chip	5
Using RAM to save variables	5
Configure digital I/O lines	6
Get switch status	6
Use high current outputs	6
Use on board opto rack	6
Connect an external opto rack	6
Learn to use G5 module	6
Use the calendar/clock	7
Write to displays	8

Also, refer to the table of contents for a listing of major 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. The following are troubleshooting hints:

1. Check the power source. If it is below 4.65 volts, the RPC-30 will be reset. Power is  $5 \pm 0.25$  volts. Make sure it is a clean 5 volt source. If it dips intermittently to 4.65 volts (due to switching noise or ripple), the card will reset for about 100 ms. If the noise is frequent enough, the card will be in permanent reset. Check U6, pin 15. If it is low (about 0 volts), then it is in reset. This line should be high (about + 5 volts).
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 v 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. The TX1 LED should flash. 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-9F. Check to make sure something is not shorting the output. It is possible for the LED to flash and not have an 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. If you are receiving a sign on message but not able to enter characters, make sure the RX1 led flashes when you hit a key.

If all of this fails, call technical support listed at the front of the book.

**INTRODUCTION**

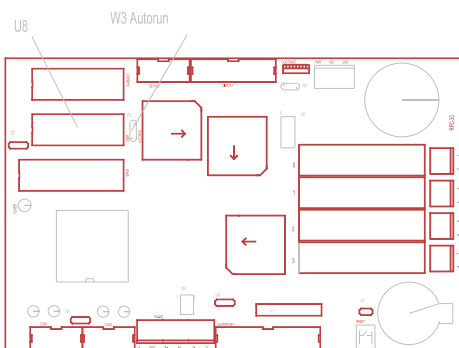
Programs are stored in EEPROM in socket U8. You can store one program up to a maximum size of about 28K bytes. A general rule to determine program storage requirements is one line requires 40 bytes. 28K bytes would store over 700 lines of code. Your application could be significantly more or less, depending upon the number of commands/line, comments, and print statements.

Despite the fact you may have a 128K or 512K RAM installed, the maximum program size CAMBASIC can run is about 28K (leaving room for variable storage). Only one program can be stored on the EEPROM. Programs cannot be chained.

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

The RPC-30 can be set to autorun on power up or reset by installing a jumper (W3). When autorun is on, the program in EEPROM is loaded into RAM and begins to execute immediately.

The RPC-30 has two EEPROMs. One is used for program storage. This is the one under discussion in this chapter. The other is a serial EEPROM used to save various CAMBASIC and user parameters and is discussed in *Chapter 13*.



**Figure 3-1 Autorun jumper**

The EEPROM 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 the serial EEPROM or RAM, which is battery backed.

**SAVING A PROGRAM**

To save a program in EEPROM, the autorun jumper W3 must be installed. Also, you need a program to save.

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 EEPROM, 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 EEPROM. If the program does not write to the EEPROM, 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 EEPROM (from above).
2. Make sure the autorun jumper W3 is 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 W3.

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 on you before putting the jumper back on.

### **LOADING AN EEPROM 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 EEPROM. 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.

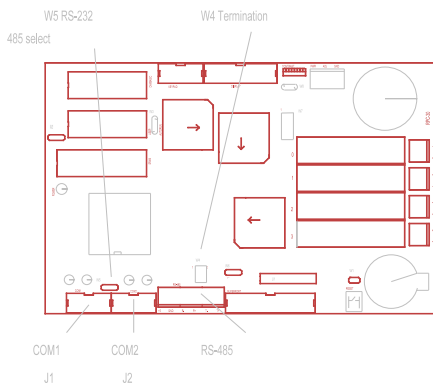
**DESCRIPTION**

The RPC-30 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 about these commands.

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**

This port uses a VTC-9F serial cable to connect external serial devices to the port. The cable consists of a 10 pin IDC connector wired one-to-one to a DB-9 connector. Line 10 is simply cut off. The pinout is designed so it plugs directly into the 9 pin serial port connector on a PC.

COM1 does not use hardware handshake lines. The CTS line is pulled high in case external equipment uses this line.

This port is normally used for programing. 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 either an RS-232 or RS-422/485 port. It also uses a VTC-9F serial cable, described above, for RS-232 level communications. COM2 is identical to COM1 except that COM2 has 2 hardware handshaking lines, CTS and RTS. When RTS goes low, the RPC-30 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(194,5)
```

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 192,4,1
500 BIT 192,4,0
```

Jumper W5 determines if COM2 is RS-232 or RS-422/485.

- [1-2] RS-485
- [2-3] RS-232 (de fault)

The CONFIG BAUD statement sets the configuration of this port.

**Termination network**

When the RPC-30 is the last physical unit on a network, or it is the only unit (RS-422), the receiver must be terminated to prevent ringing. Jumper block W4 installs or removes this network. Set W4 according to the table below:

- [1-3],[2-4] Termination network installed
- [3-5],[4-6] Termination network removed

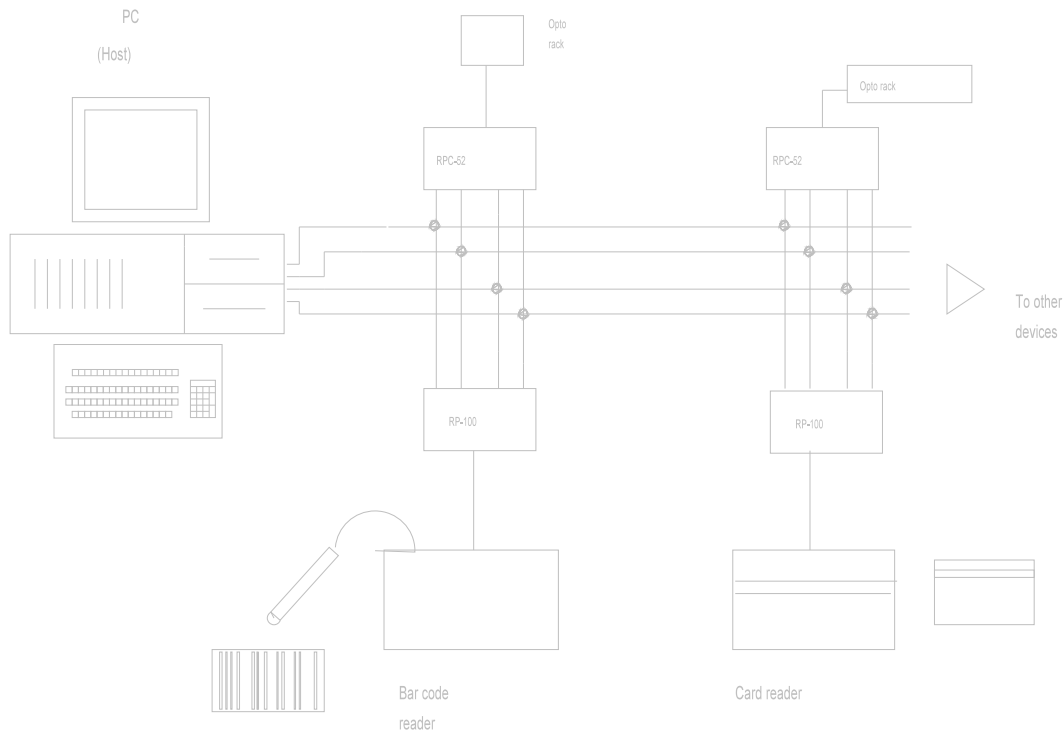


Figure 4-2 Network diagram

Only one device on a RS-485 network should have a terminator installed. The host transmitter should also have a 100 ohm resistor in series with a 0.1 mfd capacitor. The terminator on the RPC-30 includes pull up and pull down resistors to prevent lines from floating and generating erroneous characters.

**MULTIDROP NETWORK**

You can use the RPC-30 in a multidrop network by using COM2's RS-422/485 port. You can connect up to 32 units (including other RPC-30's) over a 4,000 foot range.

Figure 4-2 shows an example of a multidrop network. This network includes a host and one or more devices. The host transmits data packets to all of the devices, or nodes, in the network. A data packet includes an address, command, data, and a checksum. See figure 4-3. The packet is received by all devices, and ignored by

all except the one addressed.

The relationship described below between nodes and the host is a master-slave. The host directs all communication. Nodes "do not speak unless spoken to". Peer to peer communication, while possible with the RPC-30, is not discussed here. The problem is defining a bus arbitration scheme, of which there are many. There are many communication protocols. For this example, a protocol might look something like this:

> 22MB1

The protocol starts with the < cr> character. This character synchronizes all units and alerts them that the next few characters coming down are address and data. In this case, "> 22" is the units address. "M" is the command and "B1" is the checksum. The command is terminated with a < cr> character.

The response depends upon the nature of the command. Suppose the command M means "return a digital I/O port status". The RPC-30 could read the port and respond with AA2< cr> . The first A is an acknowledge, that is no errors were detected in the message. The data, A2, can be broken down as follows:

```
Bit/line    7 6 5 4 3 2 1 0
Status      1 0 1 0 0 0 1 0 = A2
```

Lines 1, 5 and 7 are high while the others are low.

For further information on protocols, request application note AN-2.

CAMBASIC is easily programmed as part of a network. The following code fragment shows how this is done.

```
90 CONFIG BAUD 2,6,4,0,2
100 CONFIG COM$ 2,13,20,0,0
110 ON COM$ 2 GOSUB 4000

(more program initialization here)

1000 GOTO 1000 : 'main program loop

4000 A$ = COM$(2)
4010 IF LEFT$(A$,3)<>">22" THEN RETURN
4020 B = ASC(MID$(A$,4,1))
4030 IF B = 77 THEN GOTO 4200
```

(more command parsing here)

```
4190 RETURN
```

Line 90 sets up COM 2 port for RS-485.

Line 100 and 110 set up the RPC-30 to receive characters on the RS-485 port. It instructs CAMBASIC to go to line 4000 when a < cr> or 20 characters have been received.

Line 1000 is a loop. This can be the main part of your program. Generally, non-time sensitive routines go here.

Line 4000 starts the communications task routine. This line gets the string from the buffer.

The next line, 4010, checks to see if the first three characters correspond to its address. If it doesn't, then it simply exits the subroutine. CAMBASIC requires about 3 milli-seconds to go from line 4000 to the RETURN.

4020 converts the ASCII command character to a

number. Depending upon the number of commands, it is usually faster to convert letter to a number then perform a comparison against it rather than a letter.

Line 4030 begins parsing the command to go to a particular line number.

Earlier, the amount of time to check an address and exit was given. This is important if the RPC-30 is to keep up with bus activity. This becomes a special concern when the baud rate is at 19.2K or 38.4K baud and other nodes are able to respond instantaneously to commands.

At 38.4K baud, a 7 byte packet (consisting of < cr> , address, command, and checksum) takes about 1.8 ms to send. If a node responds with a 4 byte message (acknowledge, data, and < cr> ), a complete message exchange between the host and a node took place in about 3 ms. You can see that by the time the RPC-30 determined that the incoming message was not for it, a complete exchange took place and another possibly started.

This is a problem if you are trying to run other tasks. Because new messages could be coming every 3 ms and it takes about 3 ms to determine if has been addressed, there is no time left for other tasks or running the main program.

Another time this a problem when nodes respond to messages quickly (under 100 micro-seconds), you are transmitting at 38.4K baud, and messages are always short and continuous. If the other nodes consist of RPC-30's only, then there will not be a problem.

There are a couple of solutions to this problem.

1. Lower the baud rate to 9600. Messages will now take 4 times longer.
2. Force the host to send messages every 10 ms.
3. Force nodes to delay responses to the host by 10 ms.

Solution 2 may already be enforced by the hosts processing power and programming language.



Figure 4-3 Data packet

**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 about 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" PRINT #1,"Hello" INPUT A\$ INPUT #1,A\$ A\$ = INKEY\$(1)
COM2	2	PRINT #2,"Hello" INPUT #2,A\$ A\$ = INKEY\$(2)

**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

INTRODUCTION

RPC-30 models are available with 32K or 128K of battery backed RAM. A 512K RAM may be installed at any time. RAM is in socket U10.

RAM is automatically backed up when it is installed. The battery is shared with the clock (U17) and controlled by the reset/watch dog timer (U6). Battery life will depend upon RAM size, its power consumption, and amount of time the board is operating. Generally, a battery life from 5 to 10 years can be expected.

This chapter discusses changing RAM, saving and retrieving variables, running assembly language programs, and battery maintenance. Figure 5-1 shows the location of U8, jumper W2, and the battery.

Increasing RAM size does not increase the program size CAMBASIC can handle. Maximum program and variable size is 30K. Additional RAM does increase the amount of variable storage available.

Due to the memory mapping scheme, the additional amount of memory available when a 128K or 512K RAM is installed is 32K less than the memory size. Thus, a 128K RAM provides 96K of program and data memory and a 512K provides 480K.

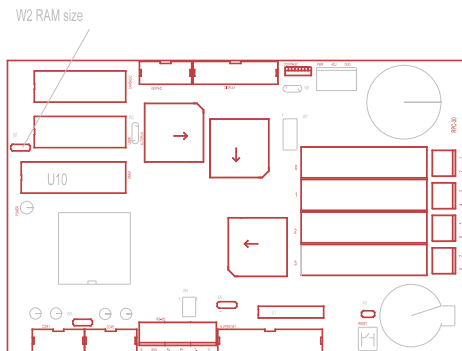


Figure 5-1 Jumper W2 and RAM chip

Both program and data are battery backed. After a power up or reset, simply type in the UNNEW command to restore the program. Variables used by the Basic program are cleared, however. Other variables can be preserved. Read RESERVING MEMORY below for more information.

There is an interaction between the speaker and when 512K of RAM is installed. This is discussed under

CHANGING MEMORY

CHANGING MEMORY

Different types of memory can be installed at any time. RPC-30 models come with either 32K or 128K of RAM installed. Up to 512K can be installed.

To change a memory chip, you need to remove the original chip, install the new one, and set jumper W2.

To install a new memory chip:

1. Turn off power to the RPC-30.
2. Remove the memory chip from U8.
3. Orient the chip so pin 1 is closest to the card edge.

If installing a 32K RAM, place the chip at the bottom of the socket (memory chip pin 14 goes into socket pin 16). The top two socket pins in each row will be empty.

If installing a 128K or 512K, install the chip into the socket.

4. Check and change, as necessary, jumper W2 to conform to the new memory.

RAM size	Jumper
32K	[2-3]
128K	[2-3]
512K	[1-2]

**NOTE:** The speaker is disabled when a 512K RAM is installed.

BATTERY BACKUP

The RPC-30 battery operates the clock and backs up the RAM when power is off. Battery life will depend upon RAM size, type, and time the RPC-30 has power applied to it. You can expect the battery to last between 3 to 10 years, depending upon operating temperature.

**NOTE:** Do not place the RPC-30 circuit on a metal surface, even with the power off, without standoffs. Voltage is present on the circuit side of the board and it is possible to short out the battery supply through the circuit traces.

Battery voltage is between 2.5 and 3.3 volts. The voltage is measured by placing a volt meter between ground and the battery clip.

The battery may be replaced by the following type or equivalent:

Panasonic BR2325

To replace the battery, lift up the holder and push the battery from behind. To install, simply reverse the procedure. The battery may be replaced while power is on. If you replace the battery with power off, be sure to reset the date and time. Also, data stored in RAM will be lost.

## STORING VARIABLES IN RAM

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

Programs and CAMBASIC variables reside in the first 64K of RAM called segment 0. Your variables are generally stored in segment 1 and higher.

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

```
20 POKE 12,A,1
```

puts the value of A into segment 1, address 12.

Use the PEEK statement to retrieve the variable:

```
50 B = PEEK(12, 1)
```

You can store and retrieve arrays, strings, and variables in this way. There are many variations of PEEK and POKE statements. Refer to the *CAMBASIC Programming Manual* for additional information and examples. A list of commands appears at the end of this chapter.

## CORRUPTED VARIABLES

The RPC-30's RAM is automatically battery backed up. User defined data can be saved when the board is powered off then on. When your application must rely on the accuracy of this data after power up, corrupted variables becomes a possibility.

The nature of RAM is it is easily written to. Any POKE'd data is susceptible to corruption. This is especially true when the board is powered down. The RPC-30 has an intelligent reset circuit which minimizes data corruption. However, when POKEing long data, such as strings and floating point numbers, a reset could interrupt a saving process. The result is information is corrupted. A scenario is explained below.

A program is running and POKEing data into RAM. At the same time it is poking, a reset occurs. A reset can occur due to power loss, someone pushing the reset button, or a watchdog timer time out.

If the program was POKEing a string (POKE\$), floating point number (FPOKE), double byte (DPOKE), or array while the reset occurred, the data became corrupted. This is because the complete value was not saved.

Since it is impossible to predict or delay a reset, a work around is to duplicate or triplicate POKEd values. That is, you would have to save the same information in two or three different places. For purposes of discussion, POKEd variables are called sets because data can consist of a mixture of variables, strings and arrays.

On power up, the program compares values from one set to the other one or two. If the two (or three) agreed, then there was no corruption and the program can reliably use the values. In practice, you would read information from set 1, but would save data to all two or three.

The use of duplicate or triplicate sets depends upon what the system must or can do if data is corrupted. When using a duplicate set, a corrupted set indicates that default values (from serial EEPROM or the program) should be used, since it is uncertain if the first or second set is corrupted. Both data sets would then be re-initialized.

A triplicate set is used to recover the last set or indicate that the data in the first set is valid. The procedure and logic is as follows.

Data is written to each element in a set in a specific and consistent order (data to an entire set does not have to be written to, just that element). For example, a calibration constant is saved (POKEd) in three different places.

Assume that the constant was assigned address 0, 100, and 200 in segment 1. The data is POKEd to address 0 first, then 100, then 200.

Upon reset, the calibration value is checked. If the value at address 0 agrees with address 100 and 200, then no corruption occurred. When address 0 and 100 agree but not 200, then this indicates that a reset occurred while updating the third set. The first data set can be trusted. The third data set simply needs to be updated.

When the first two sets do not agree, then you know that the first data is corrupted. If the second and third set agree, then, depending upon the system requirements, the first set could be "corrected" using the old data. The user or other device could be alerted that a calibration (or other) must be performed again. When all three sets disagree, then you must take action appropriate to the situation.

Another technique to check for valid memory is checksums. Simply writing a program to add the values in RAM and compare it against a number is a good check. However, you cannot tell which data element was corrupted.

Instances of data corruption are rare. They do increase as the board power is cycled or reset. You should be aware that data corruption is not impossible and there are methods to detect and correct it.

**ASSEMBLY LANGUAGE INTERFACE**

Assembly language programs (including compiled C) must start from segment 0. Use the CAMBASIC CALL statement to execute an assembly 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 lowest memory location while SYS(2) 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 an address that is 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
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 EEPROM (U8) (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. DATA statements may be entered manually or made by the MAKEDB program included with PC SmartLINK.

Line 2000 calls the program listed below. It toggles J3 line 13.

```

IN      A,(2)
LD      B,A
AND     0FEH
OUT     (2),A
LD      A,B
OR      1
OUT     (2),A
RET
    
```

**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-30 has 24 of these lines available through J3. 8 of these lines are high current outputs, capable of sinking 75 to 200 ma. Additionally, there are 4 opto-module sockets on the card itself.

On-card opto-module slots accept G4 and G5 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.

G5 series are optically isolated analog input or output modules. The modules connect to thermocouples, RTD's, load cells, 4-20 ma current loops, and general purpose voltage inputs. They can also output voltages and currents. These modules are supported by the AIN and AOT commands. Input modules return a number from 0 to 511 in a manner similar to an A-D. Conversion time is about 1 milli-second.

In addition to the 24 I/O lines from J3, the display and keypad ports can be used as digital I/O. Refer to chapters 8 and 9 for more information.

**WARNING:**

Apply power to the RPC-30 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-30 first, contact technical support for suggestions appropriate to your application.

This chapter is divided into two sections. The first section refers to the on-card opto rack. The second section refers to the digital I/O port J3.

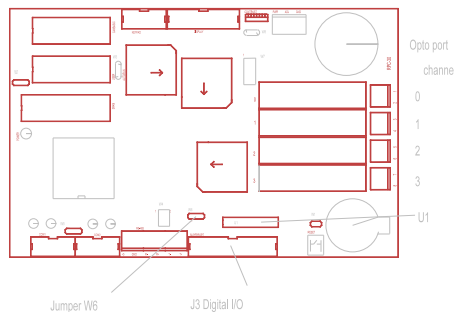


Figure 6-1 Digital I/O

ON-CARD OPTO RACK

Description

The on-card opto rack accepts the popular G4 series opto modules (manufactured by Opto-22, Grayhill, and others). These modules can switch AC or DC voltages from 12 to 240 volts at 3 amperes. They can also sense input voltages of the same type and range.

The RPC-30 also accepts the Grayhill G5 series. These modules measure voltage, current, thermocouple output, or RTD resistance and return it as a frequency. Additionally, modules can output a voltage or a current.

CAMBASIC supports the G5 series through the AIN and AOT commands.

Installation

The G4 and G5 modules are installed in the same manner as an opto rack. A screw at the top is used to secure the module to the board. Modules may be installed in any order and types can be intermixed.

A hole for a standoff near the modules is provided to keep the board from bending during installation or removal.

Input and output lines are fastened by the two position terminal in front of the appropriate opto module.

Refer to the appropriate module data sheet for additional hookup information.

G5 operation

G5 modules may be used as inputs or outputs. Output modules must be installed on an external opto rack. Its use as an input is discussed first.

Values from G5 modules are returned using the AIN function. The syntax is

$$A = \text{AIN}(\text{slot})$$

The *slot* number is from 0 to 3 or 100 to 123, corresponding to the position on the board or external opto rack. This function returns a number from 0 to 511, corresponding to a resolution of 9 bits. The AIN function takes about 1 ms to convert the input data.

The program below takes 1,000 samples and stores it in an array. The time to read and store 1,000 samples is 2 seconds, or 2 ms per sample.

```
10 DIM F(1000)
20 FOR X = 0 TO 999
30   F(X) = AIN(1)
40 NEXT
```

Some applications require that measurements be made at fixed intervals. The ON TICK construct can be used to take samples at timed intervals. The program below reads 2 channels every second and stores it into an array. When the array fills, the tick timer stops.

```
10 DIM F(800,2)
20 ON TICK 1 GOSUB 100
30 ..This is a dummy loop
40 GOTO 30
100 F(I,0) = AIN(0)
110 F(I,1) = AIN(1)
120 INC I
130 IF I = 800 THEN ON TICK 1 GOSUB 100
140 RETURN
```

The AOT statement outputs to a module. The number is in the range of 0 to 4095. The value of the output from the G5 module is in proportion to the number.

To output to a G5 module, execute the following command:

```
1000      AOT slot,value
```

*slot* ranges from 100 to 123 and corresponds to the number on an external opto rack. *value* ranges from 0 to 4095. The internal opto rack cannot be used for output.

### Converting analog measurements

Input readings can be converted to usable units of measurement by performing scaling calculations in the program. The AIN function returns values from 0 to 511. To change these reading to other units, use the following calculation:

$$\text{variable} = K * \text{AIN}(\text{slot})$$

K is a scaling constant. It is obtained by dividing the highest measurement unit number by 511.

Example:

You want to measure a 0 to 200 PSI pressure transducer with a 0 to + 5 volt output. Divide 200 by 511 to obtain the value of K.

$$K = 200 / 511$$

$$K = .3913894$$

To obtain the final value for the equation in PSI:

$$V = .3913894 * \text{AIN}(n)$$

## DIGITAL I/O PORT

Digital I/O lines on the RPC-30 is supplied by an 82C55 chip. The chip's lines go to connector J3.

Add 100 to CAMBASIC LINE and OPTO statements to access an individual port. For example, line 3 on the STB board is accessed as LINE 103, ON. Opto module position M7 is accessed as OPTO(107).

The lines on 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.

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

The address for port A is 0, B is 1, and C is 2. The CONFIG PIO statement is used to configure the 8255 lines. Use an address of 0.

Port B is connected to a high current sink through U1. See "**High current output**" later in this chapter.

### Pull up / down resistors

Digital I/O lines at J3 may be pulled up to + 5 volts or to ground through a 10K resistor pack using jumper W6.

Jumper W6 for pull up or down configuration is as follows:

- [1-2] Pull up
- [2-3] Pull down

Setting W6 for pull up makes interfacing to switches and "open collector" TTL devices easy. See "Interfacing to Switches and other devices" below.

**High current output**

Eight lines at J3 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 1.

Logic outputs 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, U1, can be replaced with a DIP shunt jumper so it is like the other lines at J3.

**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.

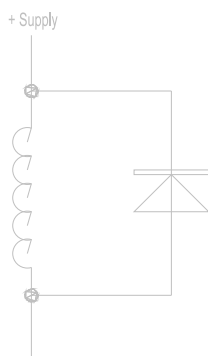
**WARNING:**

External supplies using the high current outputs must be tied to J3, 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



(to high current output)

**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**

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 and G5 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. Add 100 to any LINE command to access pins on the STB-26. For example, LINE(103) reads pin 3 at J3.

A CMA-26 connects J3 on the RPC-30 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 will cause a high voltage drop and consequently unreliable operation. Make sure you connect ground and + 5V to the opto racks.

Use the following table to determine the corresponding opto channel for a particular 82C55 port:

Opto channels	82C55 port	J3 Address
0-3	Lower C	2
4-7	Upper C	2
8-15	A	0
16-23	B	1

To turn on an opto module, a line must be low. A module is turned off by writing a '1' to a channel. The logic at port B, with the high current outputs installed is just the reverse. A '1' at a line causes the module to turn ON.

High current outputs at port B are optionally configurable as TTL I/O by replacing U1 with a DIP shunt jumper. This will keep the logic compatible with ports A and C. If opto channels 16-23 are used as inputs, then U1 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 J3 are connected to the STB-26 with a CMA-26 cable. Digital devices are then connected to the screw terminals on the STB-26.

Switches may be connected directly to a line. When jumper W6 configures the resistors as pull ups, a switch closure to ground at a line is read as a 0.

When W6 configures the input resistors as pull downs, one end of the switch must be tied to + 5 volts. If this is not possible or convenient, a 1K resistor can be tied between an input and + 5 volts to force it high when a switch is open.

**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 (J3-25), reads an opto module at channel 1 and writes an opto module at channel 5. A LED is controlled through the high current port at J3-10 (port B, bit 0).

```

200 D = BIT(0,3)      : 'Read switch status, port A
210 F = OPTO(1)       : 'read opto module, channel 1
220 OPTO 3,ON        : 'write module, channel 3
230 BIT 1,0,1        : 'turn on led at J3-10
240 BIT 1,0,0        : 'turn off led at J3-10
250 A = LINE(3)      : 'Reads pin 3 at J3
    
```

**Connector pinout - J3**

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

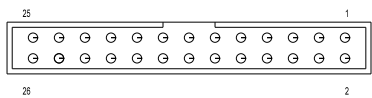


Figure 6-3 Digital I/O connector pin out (viewed from top)

**COMMANDS**

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

<u>Comm and</u>	<u>Function</u>
AIN	Returns value from G5 input module
AOT	Outputs value to G5 output module
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 an opto line
OPTO	Sets an opto module output
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-30 has a built in battery backed Calendar/clock. When used in conjunction with the DATES\$ and TIMES\$ commands, the current date and time can be set and read.

The life of the battery depends upon the power consumption of RAM in U10. Generally, you can expect a battery life of 5 to 10 years.

The clock chip, U17, contains a built in crystal. Accuracy is about 1 minute/month and is not adjustable.

Refer to the CAMBASIC Programming Manual for more specific command information.

**SETTING DATE AND TIME**

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$="03-11-93"
time$="13:56:00"
```

To retrieve date and time as part of a program:

```
2000 DA$ = DATES$(0)
2010 TI$ = TIMES$(0)
```

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

```
pr time$(0)
13:56:03
```

The clock is turned on and off and set for 12/24 hour mode using the CONFIG CLOCK statement.

**COMMANDS**

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

Command	Function
CONFIG CLOCK	Configures clock
DATES\$	Sets date
DATES\$(0)	Returns date
TIMES\$	Sets time
TIMES\$(0)	Returns time

**INTRODUCTION**

CAMBASIC and the RPC-30 can interface to a wide variety of displays:

- VF (vacuum florescent) character
- LCD (liquid crystal) character
- LCD graphics

Character display sizes range from two lines by 20 characters to four lines by 40 characters. The graphics display supports 160 x 128 pixels. Remote Processing can supply these displays with appropriate cables.

A contrast adjustment for LCD displays is built into the card. All displays connect to J4. An appropriate cable connects a display to the RPC-30.

If a display is not used, J4 may be used for general purpose digital I/O. Port A and part of port B from an 82C55 are available.

The cable length to a display depends upon the amount of current it requires. A significant amount of voltage drop occurs with a long cable. VF and LCD graphics cables should be less than 2 feet. A character LCD display cable should be less than 5 feet.

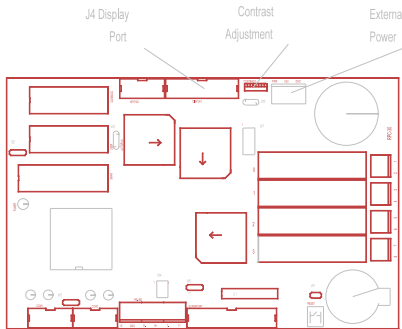


Figure 8-1 Display port

**CONNECTING DISPLAYS**

The display port is designed to supply all the lines necessary for VF and LCD displays. A custom cable connects the RPC-30 to the display.

Displays purchased from Remote Processing include a

cable. You simply connect the 20 pin connector to the RPC-30 LCD display port and the other end into the display.

Additional wiring is usually required for LCD graphic and VF character displays. This information is included with the display. Information content is display dependent. Below is general information on both.

Graphic displays require additional voltages not generated on the RPC-30. These must be supplied externally. An external contrast adjustment may be necessary.

**WARNING:**

*PWR* and *ADJ* silkscreen on the RPC-30 card are reversed. Apply power to the terminal marked *ADJ* (DP and LCD-5003). Connect the wiper from the contrast adjust to the terminal marked *PWR* (for LCD-5003 or other graphic displays only).

VF character displays require + 5 volts and ground to be brought to connector P6. This may be in the form of external wires from the main power connector on the board. The reasons power is not supplied are:

1. The display requires 400-600 ma for operation. By switching voltage through an opto module, power to the display is controlled.
2. Since power is so great, there is the chance for ground loops on the card.

**WRITING TO THE DISPLAY**

CAMBASIC's DISPLAY command is used to print information. The display type is set by the CONFIG DISPLAY command.

**PROGRAMMING EXAMPLE**

The example below is for a four line by 40 character LCD display. Notice that all DISPLAY statements end with a semicolon (;) so a < cr> < lf> sequence is not sent. Doing so on LCD displays prints "special" or unpredictable characters.

```
10 CONFIG DISPLAY &50,7,1
20 A$ = "Remote Processing LCD display"
30 DISPLAY (0,5) A$;
40 DISPLAY (1,10) "with LED backlight";
```

**DISPLAY TYPES**

CAMBASIC assumes there is a controller on the display. Its software driver is based upon the characteristics of the display family. Compatible VF and LCD displays are shown below:

Manu fact.	Model	Type
Optrex	DMC 40457	LCD 4 x 40
Optrex	DMC 40202	LCD 2 x 40
IEEE	3601-90-080	DF 4 x 20
Optrex	DMF 5003N	LCD 160 x 128

**DISPLAY CONNECTOR PINOUT - J4**

The display port uses an 82C55 for data and control. The table below lists a pin number and its intended function. A display may not use all lines even though they are available.

J4	8255	Function
Pin	Port/line	
1		Logic + 5V
2		Digital ground
3	A/4	D4
4		Contrast
5	A/6	D6
6	A/5	D5
7	B/4	Reset
8	B/3	Write
9	B/2	Read
10	A/7	D7
11	A/1	D1
12	A/0	D0
13	A/3	D3
14	A/2	D2
15	B/7	CS
16	B/6	Command/ data
17	B/5	Halt
18		Ext. contrast
19		Alternate power
20		Power ground

The contrast adjustment (J4-4) provides a variable voltage from + 5 to -7 VDC. External contrast (J4-18) is an external voltage required for graphics displays.

J4 is available for additional I/O if a display is not used. Port A may be configured as an input or output. Port B must be configured as an output if a 17 key or larger keypad is used.

The address for port A is 80; port B is 81. Use CONFIG PIO to configure the port.

**COMMANDS**

The following CAMBASIC commands are used for the display.

Command	Function
CONFIG DISPLAY	Specifies the display type to use. Use & 50 for the address.
DISPLAY	Prints the string at the row and collum specified.

The following commands are applicable to the LCD-5003 graphics display.

DISPLAY LINE	Draws a line.
DISPLAY P	Prints a dot to the screen.
DISPLAY [ON][OFF]	Turns on graphics and character displays.
CLEAR DISPLAY	Clears graphics and/or character display.

See the DISPLAY command for more variations.

## INTRODUCTION

16 to 24 position keypads are plugged into keypad port J5. Keys are arranged in a 4 x 4 to 4 x 6 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 24 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.

The 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.

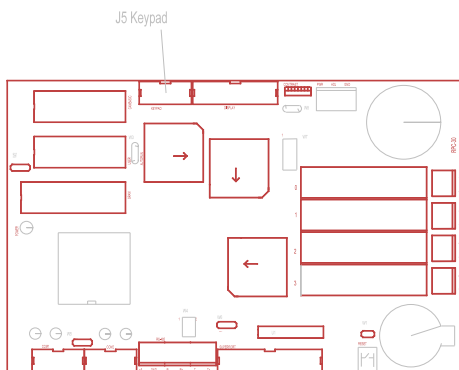


Figure 9-1 Keypad connector

## 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.

```

20 'Optionally change keypad character 'B'
30 ' to the letter 'M'
40 POKE SYS(8)+7,77
50 CONFIG DISPLAY &50,7,1
60 ON KEYPAD$ GOSUB 500
70 DISPLAY (0,0)"Enter a number"
80 DISPLAY (1,0)
100 'loop for this example
110 GOTO 100

500 A$ = KEYPAD$(0)

```

```

510 IF A$ = "C" THEN ..clear_bEEP
520 IF A$ = "#" THEN ..enter
530 DISPLAY A$;
540 B$ = B$+A$
550 SOUND 600,.2
560 RETURN

```

```

600 ..clear_bEEP
610 B$=""
620 SOUND 600,.2
630 DELAY .4
640 SOUND 600,.2
650 DISPLAY (1,0) "          ";
660 RETURN

```

```

700 ..enter
710 FL = 1
720 SOUND 800,.5
730 RETURN

```

### Program explanation

Lines 10-80 set up the parameters for the keypad and display. Line 80 positions the cursor on the next line.

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. A beep indicates it was saved.

Lines 600 to 660 send a double beep when data is cleared. The buffer (B\$) is cleared as is the display.

Lines 700 to 730 process the "enter" key. A longer beep is sounded and 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. Keypad positions 17-24 do not have a character assigned to them. These positions are assigned using the SYS(8) statement.

## KEYPAD PORT PINOUT - J5

The keypad port uses ports B and C from an 82C55. Lower port C is configured as an input. Upper port C and port B bits 0 and 1 are outputs. Port B bits are used only when a 20 or 24 position keypad is installed.

The table below lists J5's pinout, 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	B/0	Column 5
10	B/1	Column 6

## 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**

A modem quality speaker is installed on the circuit board. Useful frequency range is 200 to 4000 hz. Both frequency and duration are set by the SOUND command.

The speaker must be disabled when a 512K RAM is installed. This is due to the fact one of the address lines is also used as a frequency output.

The AIN command also shuts off the SOUND.

Speaker output is controlled by jumper W8.

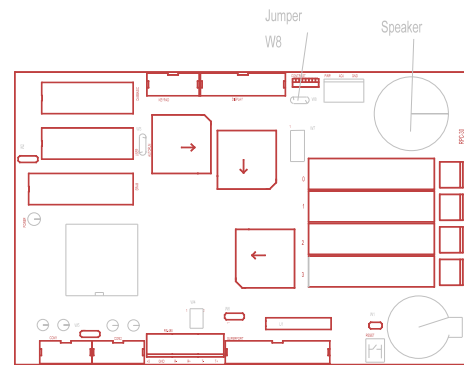
- [1-2] Speaker disabled, 512K RAM installed
- [2-3] Speaker enabled, < 512K RAM installed

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.



**Figure 10-1 Jumper W8 and speaker**

The AIN and AOT commands will shut off the sound.

**NOTE:** When SOUND is used with a time parameter, program execution is suspended until it is timed out. Executing SOUND with a time parameter has the same effect on a program as executing the DELAY statement.

**DESCRIPTION**

The watchdog timer is used to reset the RPC-30 if the program or CPU "crashes". When enabled, the program must write and data to I/O address 96 at least once every 1.2 seconds to avoid a reset. The timeout is not adjustable.

The watchdog should be disabled when using INPUT and DELAY statements. Also, loops which do not end quickly or are of indeterminate duration should be avoided unless a timer reset pulse is included. An example of an indeterminate loop is one that waits for a port condition to change.

The watchdog is enabled by writing a 1 to address 16, bit 1 and disabled by writing a 0 to the same location (use the BIT statement to do this). The timer is reset by a write of any data to I/O address 0.

The watchdog timer is part of a voltage monitor, battery backup controller, and reset chip U6.

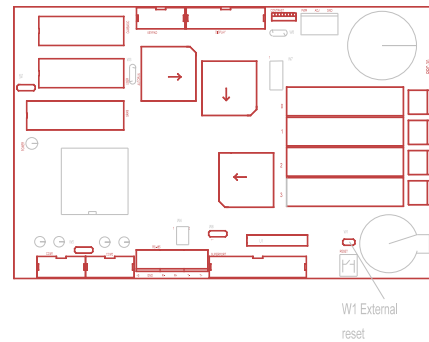
**PROGRAM EXAMPLES**

The following program fragments enable the watchdog timer, reset it while the program is running, and then disables it.

```
100 BIT 16,1,1 :'Turn on watchdog
.
.
.
5000 OUT 96,0 :'Reset timer
.
.
.
10000 BIT 16,1,0 :'Turn off watchdog
```

**EXTERNAL RESET**

The RPC-30 may be reset by an external switch via jumper W1. Short terminals 1 to 2 to cause a reset. If the reset line is excessively long or in an electrically noisy environment, connect a 1K pull-up resistor to W1-1. This will prevent erroneous resets.

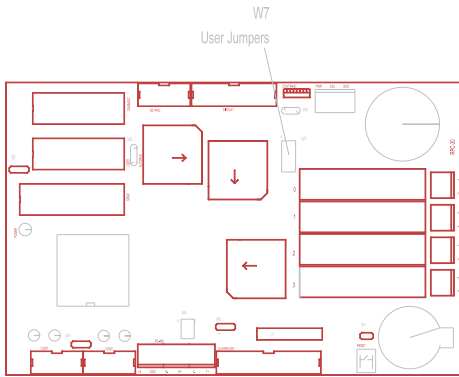


**Figure 11-1 External reset**

**DESCRIPTION**

Five user jumpers are available at W7. These jumpers may be read as part of a program to determine a boards function or configuration. It is up to you to determine what the jumpers mean. A common use is to set the boards address in a RS-485 network.

Jumpers are at address 17, bits 1-5. Bit 0 is used by the system to read the serial EEPROM.



**Figure 12-1 W7 user jumpers**

Jumper W 7 is mapped to the following bit numbers.

Jumper	Bit No.
[1-2]	5
[3-4]	4
[5-6]	3
[7-8]	2
[9-10]	1

Individual jumper status are read using the BIT function. A '0' indicates a jumper is installed.

```
100 A = BIT(17,3) : 'Reads W7[5-6]
```

Jumpers may be read to determine a card address. Use the following map to assign a value to a jumper. When a jumper is installed, its value is 0. When removed, its value is shown to the right. The program example shows how a jumper configuration can be converted into a number and a string.

Jumper	Value
[9-10]	1
[7-8]	2
[5-6]	4
[3-4]	8
[1-2]	16

Assume W7[7-8] and [1-2] are jumpered. The program would return the following value.

```
10 A = INP(17)/2 : 'Get jumper value & shift
20 A$ = STR$(A) : 'Convert to a string
30 A$ = RIGHT$(A$,LEN(A$)-1) : 'strip off space
40 PRINT "Jumper value is:";
50 PRINT "Jumper string is:";A$
RUN
Jumper value is: 18
Jumper string is:18
```

Line 30 strips off the leading space when a number is converted to a string.

This string could be concatenated to produce a board identifier. See Chapter 4, Networking, for more information.

**DESCRIPTION**

The serial EEPROM is a 128 byte, non-volatile device that stores your programs parameters. Strings, integers, and floating point numbers can be saved to EEPROM. Information such as calibration constants, recipes, RS-485 address, or other "soft" information that may change over time should be stored in the serial EEPROM.

The LOAD # and SAVE # commands are used to get and store data to and from the EEPROM. The syntax is:

```
SAVE #address
LOAD #address
```

Where address is the location in RAM. address is in the range of &8000 to &FFFF. Take care to make sure you are not POKEing into program or reserved memory addresses. Use the SYS(1) and SYS(2) functions to the addresses for free space. LOAD and SAVE effectively require 128 bytes of free RAM. The LOAD # command reads 128 bytes from the serial EEPROM and transfers this information starting at the address specified.

An EEPROM is more secure than battery backed RAM because it is more difficult to write to it. Several microprocessor instructions must take place before a byte is changed. RAM, on the other hand, requires only a momentary pulse to modify its memory.

Each byte can be written to 10,000 times and read from any number of times. The EEPROM could be updated once a day for over 27 years before this limit is exceeded.

Do not constantly store information to the EEPROM. That is, do not continuously write to it once a second as part of your program. This is one electronic part you can "wear out".

**PROGRAM EXAMPLE**

The following program example saves and retrieves a string and a number to the serial EEPROM.

```
100 'Save to EEPROM
110   'Put string of known length to RAM
120   A$ = ">03"
130   C = 14.35
140   POKE$ &E000,A$
150   'Put a number to RAM
160   FPOKE &E020,C
170 'Save to EEPROM
180   SAVE #&E000

200 'Load from EEPROM
210   LOAD #&E000
220   'Now put info in variables
230   B$ = PEEK$(&E000)
240   D = FPEEK(&E000)
```

When saving strings, the amount of memory required is 1 + the string length. Floating point numbers require 6 bytes, integers require only 1 byte. You may find it desirable to make a "memory" map of the data types you wish to save. You can "stack" different kinds of data next to each other (strings, integers, floating point numbers) so long as you know where you are saving and retrieving them.

## ELECTRICAL

### CPU

Z180, 18 Mhz clock

### Memory

CAMBASIC, 32K ROM

Programming and data is 32K or 128K RAM standard, 512K Optional.

**RAM** is battery backed up. Battery life is 5-10 years depending upon RAM size, type, and operating time.

Program is 32K EEPROM

Memory speeds are 80 ns or faster

### Digital I/O

The RPC-30 has 48 digital I/O lines. 24 are from J3, which is a general purpose port. 14 are for the display port, J4. 10 are for the keypad port, J5. All ports use an 82C55 for interfacing.

The specifications below are for all digital I/O except for the eight high current lines at J3.

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 J3

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

### Keypad input

10 lines accept a 16, 20, or 24 position matrix keypad. Scanning and debounce performed in CAMBASIC.

### Display output

14 digital and 6 power and ground lines used to control LCD, VF, and LCD graphics displays. Displays supported 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 lines. COM2 configurable to RS-232 or RS-422/485. Termination network for RS-422/485 available. Baud rates from 600 to 38.4K, 7 or 8 data bits, parity even, odd, or none, 1 or 2 stop bits.

### EEPROM and programmer

Accepts 29C256 or equivalent EEPROM.

Size:32K

Speed:120ns or faster.

Opto module rack

Four position accepts G4 or G5 series I/O modules

### Calendar/Clock

Accuracy to 1 minute/month

Supported by CAMBASIC.

Battery backup standard. Expected life 5 to 10 years depending upon RAM installed and operating time.

### Watchdog timer, reset

Watch dog timer resets card for 150 ms minimum when enabled.

Reset connector available. External reset time should be 0.1 second minimum.

Push button reset included.

### Power requirements

+ 5 ±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 7.0"

Maximum height: 0.675", no opto modules or cables installed.

Mounting holes: 4 each corner, 0.25" from edge of board. Hole size is 0.156" dia.

# TECHNICAL INFORMATION

## MEMORY AND I/O MAP

### Memory

Description		Address
CAMBASIC	U9	&00000 - &07FFF
RAM, U10	32K	&08000 - &0FFFF
	128K	&08000 - &1FFFF
	512K	&08000 - &7FFFF

### I/O

J3 Digital	&0000 - &000F
On card opto modules, jumpers, watchdog	&0010 - &001F
Display & keypad	&0050 - &005F
Watchdog timer reset	&0060 - &006F
Real time clock	&0070 - &007F
Internal processor	&0080 - &00FF
Program EEPROM	&8000 - &FFFF

## JUMPER DESCRIPTIONS

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

Jumper	Description
W1[1-2]	External reset
W2[1-2]	A17 for 512K RAM
W2[2-3]*	VBAT for 32K & 128K RAM
W3[1-2]*	Autorun
W4[3-1]	RS-485 terminated
W4[3-2]*	RS-485 not terminated
W4[4-2]	RS-485 terminated
W4[4-6]*	RS-485 not terminated
W5[1-2]	COM2 is RS-422/485
W5[2-3]*	COM2 is RS-232
W6[1-2]*	J3 resistors pulled up
W6[2-3]	J3 resistors pulled down
W7[1-2]*	User jumper 5
W7[3-4]*	User jumper 4
W7[5-6]*	User jumper 3
W7[7-8]	User jumper 2
W7[9-10]	User jumper 1
W8[1-2]	Speaker off
W8[2-3]*	Speaker enabled