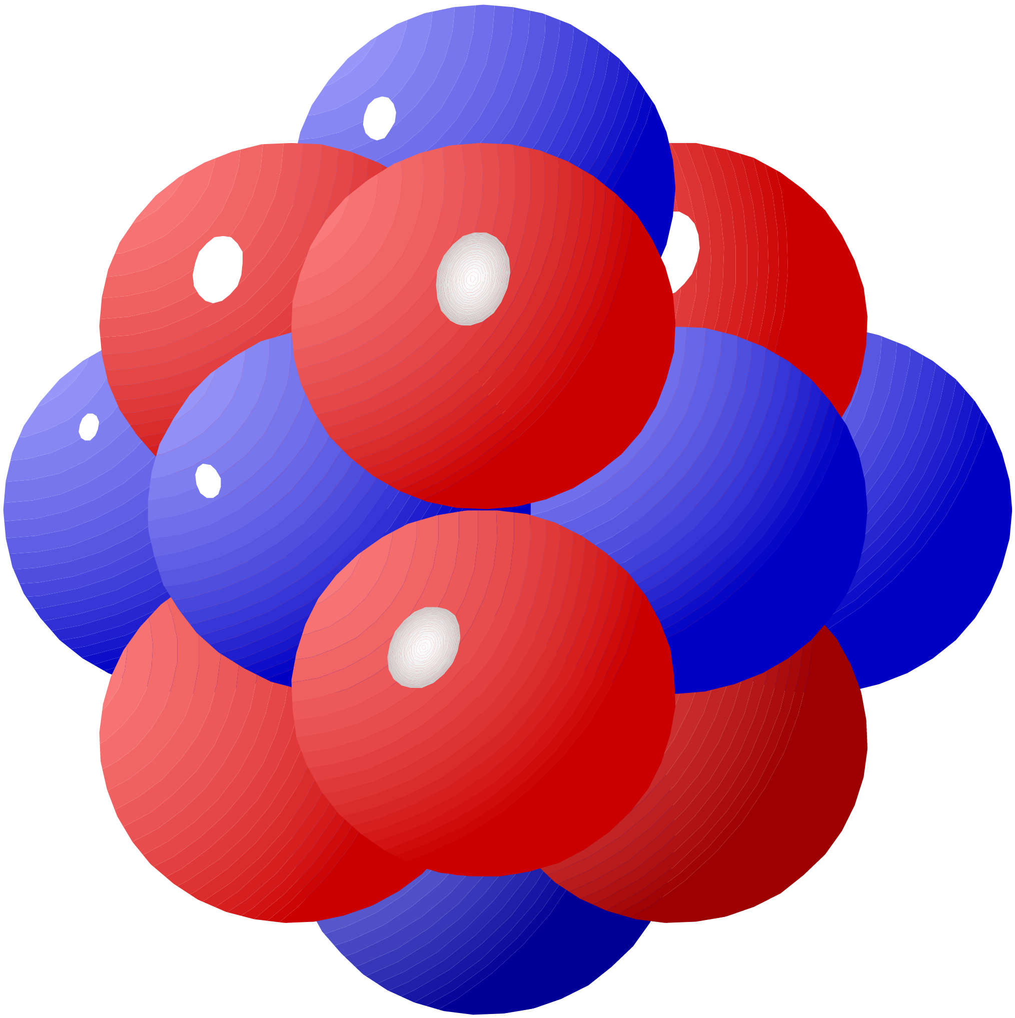
PROTON

Serial LCD.



Crownhill Associates
smart electronic solutions

Disclaimer

In order to comply with EMC directive 89/336/EEC, this product should not be used outside of a classroom or laboratory environment.

Any software supplied with this product is provided in an “as is” condition. No warranties, whether express, implied or statutory, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose apply to this software. The company shall not, in any circumstances, be liable for special, incidental or consequential damages, for any reason whatsoever.

This product should not be used as a critical component in life support systems. Crownhill does not recommend the use of any of its products in life support applications where the failure or malfunction of the product can reasonably be expected to cause failure in the life support system or to significantly affect its safety or effectiveness.

In producing this, document and the associated hardware we have spent a great deal of time to ensure the accuracy of the information presented. We constantly strive to improve the quality of our products and documentation. Crownhill operates a quality system conforming to the requirements of BS EN ISO9001: 2000. Crownhill reserves the right to change the functionality and specifications of its products from time to time without prior notice.

If you should find any errors or omission in these documents or hardware, please contact us, we appreciate your assistance in improving our products and services.

Crownhill reserves the right to make changes to the products contained in this publication in order to improve design, performance or reliability. Crownhill assumes no responsibility for the use of any circuits described herein, conveys no license under any patent or other right, and makes no representation that the circuits are free of patent infringement. Charts and schedules contained herein reflect representative operating parameters, and may vary depending upon a user’s specific application. While the information in this publication has been carefully checked, Crownhill shall not be liable for any damages arising as a result of any error or omission.

PICmicro^{tm} is a trade name of Microchip Technologies Inc.
PROTON^{tm} is a trade name of Crownhill Associates Ltd.
EPIC^{tm} is a trade name of microEngineering Labs Inc.

This document was written by Les Johnson and published by Crownhill associates limited, Cambridge England, 2003

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

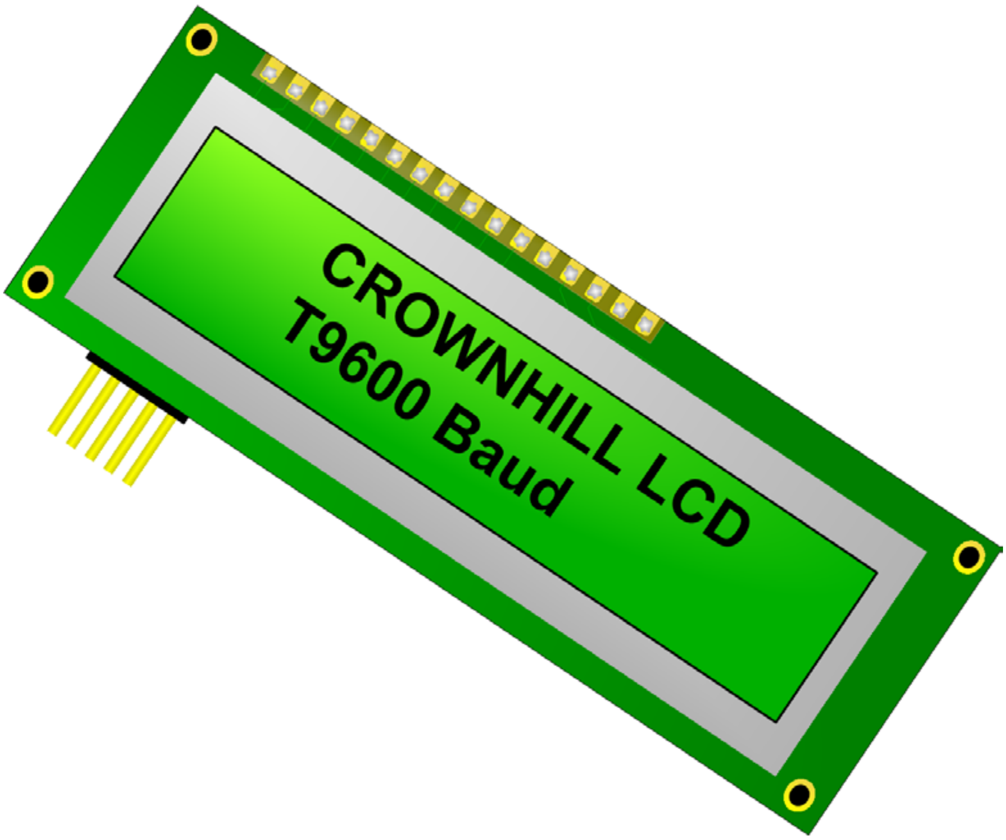
Introduction

LCD displays are an invaluable aid to programming, as well as offering a visual interface to the outside world of humans. However, they do tend to use up precious pins on the PICmicro, which makes them prohibitive on smaller devices.

The PROTON Serial LCD gives the ability to interface to an LCD with just 1 pin, using serial RS232 data to give instructions to an on-board microcontroller which actually controls the LCD.

If used in conjunction with the PROTON or PROTON+ Compilers, the LCD may be treat as if it were attached as normal, but instead of using the PRINT command, one of the host of serial commands may be used instead, such as RSOUT.

One of the most useful features of the PROTON Serial LCD is its ability to be re-programmed with code of your own creation. Be it a customised serial interface, a frequency meter, an event counter, or a whole host of other devices that require a microcontrolled LCD with some I/O available.



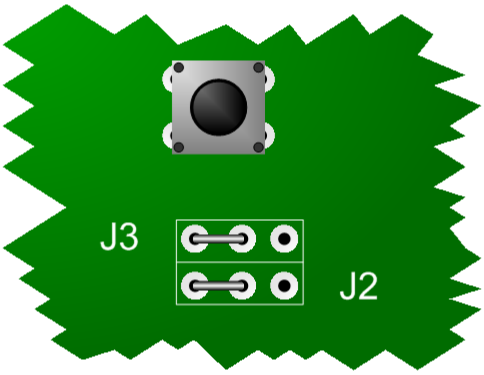
Copyright © Crownhill Associates Limited 2003
All Rights Reserved

Configuring the Serial Interface.

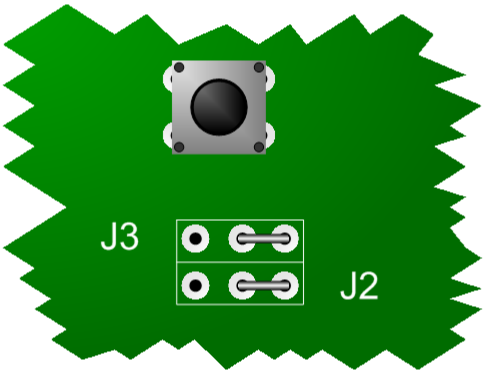
There are many forms of RS232 data structures, including 7-bit data, 2 stop bits, and parity checks, and all these can be catered for by creating your own code for the serial interface. However, the default firmware included with the PROTON serial LCD uses No Parity, 1 start bit, 8 data bits, and 1 stop bit (8N1), and the ability to change the serial mode from INVERTED to TRUE or vise versa.

Inverted and True modes offer the ability to interface directly to the PC (True Mode), or directly to the PICmicro's USART (Inverted Mode). The choice of the mode used depends on the application it is interfacing with.

Two jumpers are used to change the serial mode. These are J2 and J3 shown below.



TRUE Serial Mode



INVERTED Serial Mode

Note that the RESET button is not fitted as standard to the PROTON Serial LCD, and is only shown in the above diagrams as a reference point. The reset button may be added by the user if required.

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

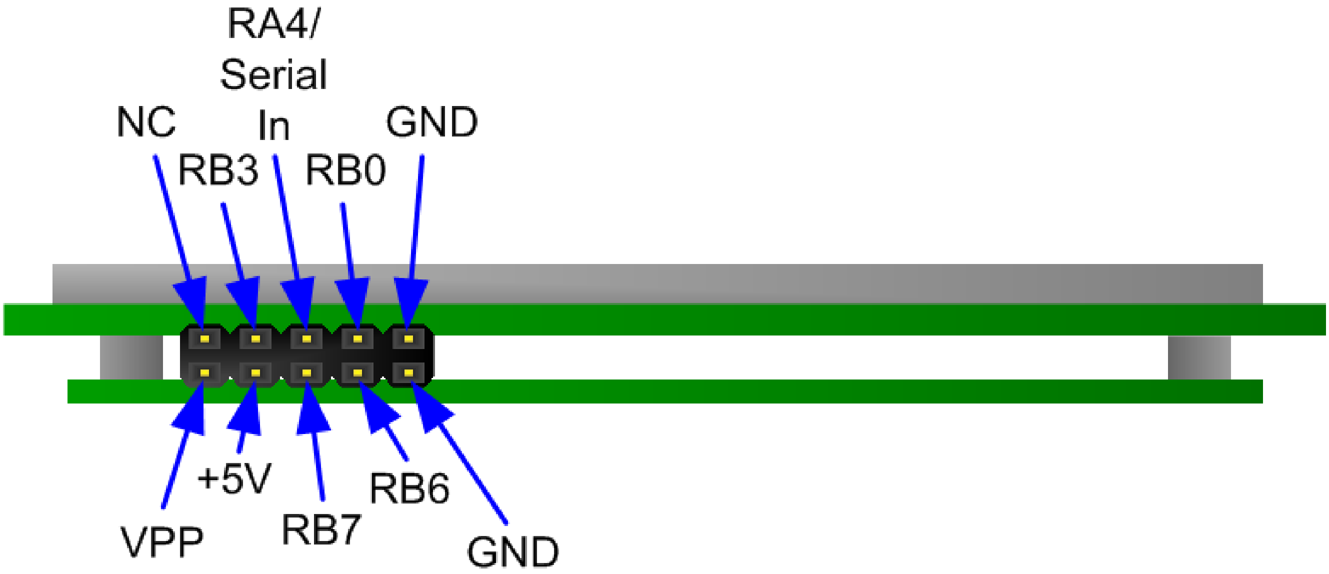
Jumper J3 connects to pin RB4 of the PICmicro^{tm}, and is used to detect the required mode in software. Jumper J3 is used to lightly pull up or down the serial data pin (RA4), which stops any spurious data being displayed when the serial in line is left floating. i.e. not connected to anything.

Each mode (inverted or true) requires a different polarity weak pull on the serial in line.

Physical Connections.

Connecting the outside world to the PROTON serial LCD is accomplished through a 10-way header. This not only carries the serial data, but also supplies the 5 Volts required by the circuit, and offers a method of re-programming the on-board 16F84A PICmicro^{tm}. Two extra pins are also brought out of the header that may be used in your own code creations. These are RB0, and RB3.

The diagram below shows the pinouts of the 10-way header, and their connections to the PROTON serial LCD's circuit.



Pins RB6, RB7, and VPP are used for In-Circuit-Serial-Programming (ICSP) of the on-board 16F84A PICmicro^{tm}, but RB6, and RB7 can also be used as general purpose I/O lines when creating your own code.

Baud Rate Adjustment.

The serial interface initially requires TRUE 9600 baud serial data for its transfers, but lower baud rates can be realised via software.

Negotiating the baud rate used by the serial LCD controller is both simple in its design, and efficient in its usage.

Issue the standard control value of 254, then value 253, then the baud rate (in ASCII) desired (up to 9600 baud), terminated by a carriage return (13). For example, to change the baud rate from 9600 to 4800, the following line of code could be used: -

```
Serout GPIO.2,T9600,[254,253,"4800",13] ' Set baud rate to 4800
```

The LCD controller will acknowledge the change of baud by transmitting the letter "O" for OK or "E" for an invalid baud rate value; at the original baud rate. This can be tracked by the program by using the following lines of code: -

```
TRY_AGAIN:
Serin GPIO.2,T9600,1000,TRY_AGAIN,[ACK_BYTE] ' Receive the ACK byte
If ACK_BYTE <> "O" Then   ' Have we received the ACK byte ?
     Goto TRY_AGAIN       ' No. So try again
Else                      ' Otherwise
     Delayms 100          ' Wait for the controller to adjust itself
```

At higher baud rates, a delay may need to be placed before each character by using the **RSOUT_PACE DECLARE** if using the **RSOUT** command, or using the PACE operand if using **SEROUT**. This allow the serial LCD controller to reliably capture all commands and data.

```
Serout GPIO.2,T9600,5,["HELLO WORLD"]
```

The line of code above will place a 5ms delay between the characters of the message "HELLO WORLD".

In operational tests carried out, a delay between characters was not generally required, but you should be aware that it may be.

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

A demonstration program showing the use of baud rate changes is shown below.

```
' Negotiate three baud rates on the PROTON Serial LCD
' Using a 12F675 8-pin PICmicro

Device = 12F675
XTAL = 4

Dim ACK_BYTE as Byte

Symbol T1200 = 813
Symbol T4800 = 188
Symbol T9600 = 84

Delayms 500                  ' Wait for PICmicro to stabilise
ANSEL = 0                    ' Set pins to digital mode
CMCON = 7                    ' Disable ADCs
Serout GPIO.2, T9600,[Cls,"HELLO THERE"]
Serout GPIO.2, T9600,[254,192,"AT 9600 BAUD"]
Delayms 500                  ' Display for 500ms

Serout GPIO.2, T9600,[254,253,"4800",13]     ' Set BAUD rate to 4800
' Receive the ACK byte
TRY_AGAIN_FOR_4800:
Serin GPIO.2, T9600,1000,TRY_AGAIN_FOR_4800,[ACK_BYTE]
If ACK_BYTE <> "O" Then TRY_AGAIN_FOR_4800 : Else : Delayms 100

Serout GPIO.2,T4800,[Cls,"HELLO AGAIN"]
Serout GPIO.2,T4800,[254,192,"AT 4800 BAUD"]
Delayms 500                  ' Display for 500ms

Serout GPIO.2,T4800,[254,253,"1200",13] ' Set BAUD rate to 1200
' Receive the ACK byte
TRY_AGAIN_FOR_1200:
Serin GPIO.2,T4800,1000,TRY_AGAIN_FOR_1200,[ACK_BYTE]
If ACK_BYTE <> "O" Then TRY_AGAIN_FOR_1200 : Else : Delayms 100

Serout GPIO.2,T1200,[Cls,"HELLO AGAIN"]
Serout GPIO.2,T1200,[254,192,"AT 1200 BAUD"]
Stop
```

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

A demonstration program showing the general use of the Serial LCD is listed below. The code is for use with an 8-pin 12F675 device.

```
' Write the text 'HELLO WORLD' on the LCD
' Using a 12F675 8-pin PICmicro

Device = 12F675
XTAL = 4

RSOUT_PIN = GPIO.2          ' Serial out pin
RSOUT_MODE = TRUE           ' Set TRUE mode
SERIAL_BAUD = 9600          ' Baud rate of 9600
RSOUT_PACE = 1              ' 1ms delay between characters

Dim COUNT_VARIABLE as DWord

Delayms 500                 ' Wait for PICmicro to stabilise
ANSEL = 0                   ' Set pins to digital mode
CMCON = 7                   ' Disable ADCs
Rsout Cls                   ' Clear the LCD

Rsout "HELLO WORLD"         ' Write the text
AGAIN:
For COUNT_VARIABLE = 0 to 2000000
  Rsout at 2,1,DEC COUNT_VARIABLE,"          "
  Delayms 200
Next
Goto AGAIN
```

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

If you've re-programmed the Serial LCD controller board with a piece of software of your own creation, you may wish to replace the original code at a later date, so it is presented in full below.

```
' Program SERIAL_INTERFACE.BAS
'
' Serial LCD interface for 16F84A PICmicro device
'
' Inludes baud rate change with control codes: -
' 254,253,"BAUD RATE IN ASCII"
'
' First Revision Written 30-5-2003

Device = 16F84A                    ' Target device is 16F84A
XTAL = 8                           ' Use an 8MHz crystal

LCD_DTPIN = PORTA.0                ' DATA pins start at PORTA.0
LCD_RSPIN = PORTB.2                ' RS pin on PORTB.2
LCD_ENPIN = PORTB.1                ' EN pin on PORTB.1
LCD_INTERFACE = 4                  ' 4-bit Interface
LCD_LINES = 2                      ' 2 line LCD type

Symbol T9600 = 84        ' Value used by SERIN for True 9600 baud
Symbol N9600 = 16468 ' Value used by SERIN for Inverted 9600 baud
Dim DATA_BYTE as Byte    ' Contains byte received form serial pin
Dim BAUD_IN as Word ' Contains baud rate received from serial pin
Dim TEMP_DWORD as Dword            ' Temporary 32-bit variable
Dim BAUDRATE as Word               ' Contains the current baud rate
Dim OLD_BAUDRATE as Word           ' Contains the old baud rate

Symbol MODE_DETECT = PORTB.4  ' Detects serial mode for interface
Symbol RX_PIN = PORTA.4            ' Serial IN pin
Symbol TX_PIN = PORTA.4            ' Serial OUT pin

'----{INITIAL SETUP ROUTINE]----------------------------------------
Delayms 100                        ' Wait for PICmicro to stabilise
Cls                                ' Clear the LCD
Print at 1,2,"CROWNHILL LCD"
Input MODE_DETECT        ' Set the MODE detection pin as an INPUT
If MODE_DETECT = 0 Then            ' Monitor the MODE DETECT pin
 BAUDRATE = T9600                  ' Default baud rate is TRUE mode
 OLD_BAUDRATE = T9600              ' ditto
 Print at 2,2,"T9600 Baud"
Else
 BAUDRATE = N9600             ' Default baud rate is INVERTED mode
 OLD_BAUDRATE = N9600              ' ditto
 Print at 2,2,"N9600 Baud"
Endif
Goto MAIN_PROGRAM                  ' Jump to main program loop
```

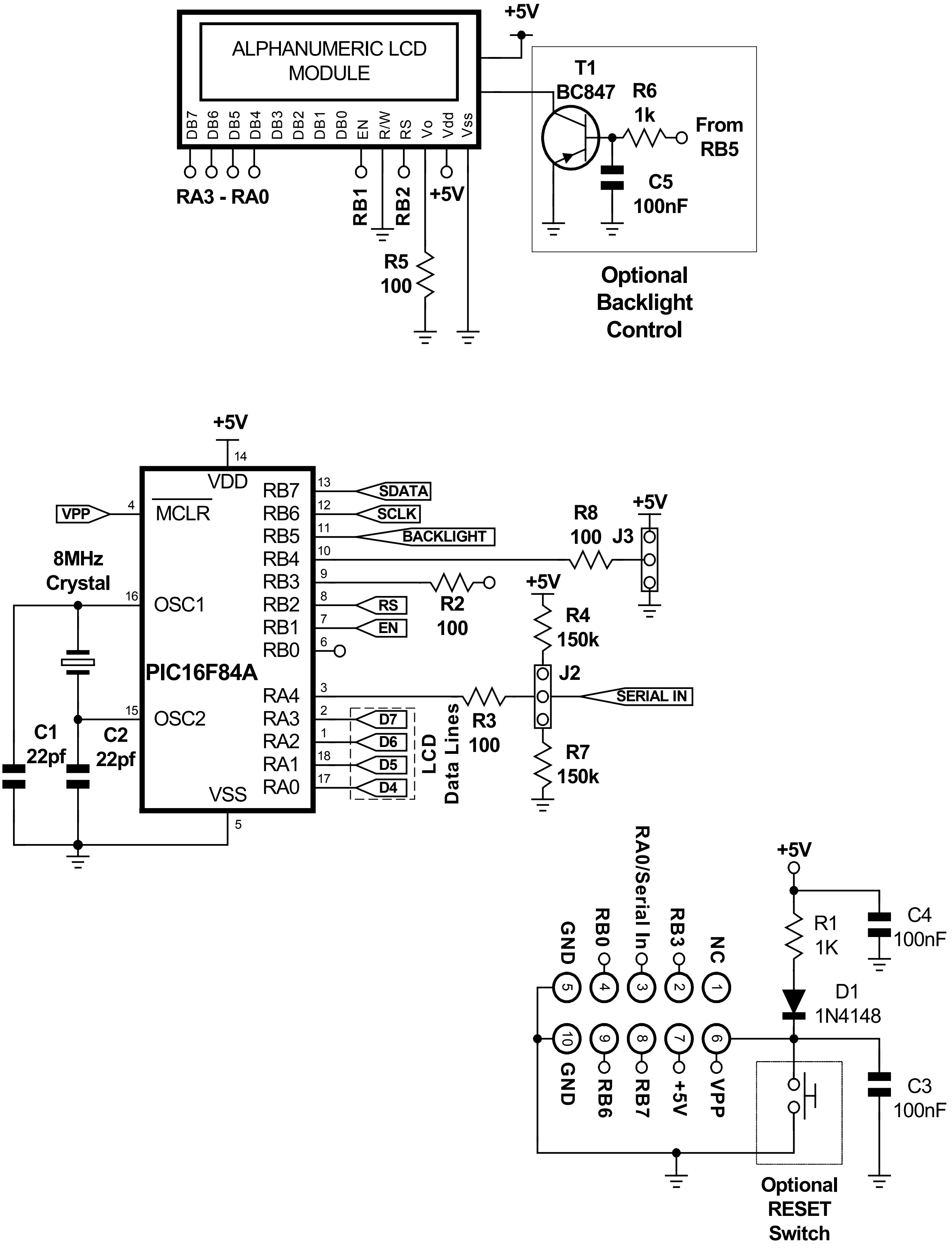
```
'----[BAUD CHANGE ROUTINE]------------------------------------------
' Receive the new baud rate.
' And calculate value required for SERIN.
' See the manual or help file for the specific calculation.
' Returns an ACK of "O" if OK.
' Returns an ACK of "E" if baud rate out of range.
CHANGE_BAUD:
SERIN RX_PIN,BAUDRATE,[DEC BAUD_IN]     ' Receive the new baud rate
If BAUD_IN = 0 OR BAUD_IN > 9600 Then ' Is baud rate within limits ?
 Delayms 30                             ' No.. So wait 30ms
 SEROUT TX_PIN,BAUDRATE,["E"]      ' And send an ERROR Acknowledge
Goto MAIN_PROGRAM                  ' Look for another serial byte
Endif
TEMP_DWORD  = 1000000 / BAUD_IN    ' 32-bit calculation
BAUDRATE = TEMP_DWORD - 20         ' Calculate the new baud rate
If MODE_DETECT = 1 Then BAUDRATE.14 = 1 ' Set to inverted mode ?
Delayms 30                         ' Wait 30ms
' Send an Acknowledge at the old baud rate
SEROUT TX_PIN,OLD_BAUDRATE,["O"]
OLD_BAUDRATE = BAUDRATE  ' Change the old baud rate to the new rate
' Fall through to the main program loop

'----[MAIN PROGRAM LOOP STARTS HERE]--------------------------------

MAIN_PROGRAM:
While 1 = 1                        ' Create an infinite loop
 SERIN RX_PIN,BAUDRATE,[DATA_BYTE] ' Receive a byte from serial pin
 If DATA_BYTE.7 = 1 Then      ' Is control bit (bit-7) set. i.e. 254
  Call SERIN                  ' Receive another command quickly
  Movwf DATA_BYTE             ' Transfer WREG to DATA_BYTE
  ' Jump if BAUD RATE change raquired
  If DATA_BYTE = 253 Then CHANGE_BAUD
  Print 254                   ' Otherwise, send a control byte
 Endif
 Print DATA_BYTE              ' Then Send data straight
Wend                          ' Loop forever
```

Copyright © Crownhill Associates Limited 2003
All Rights Reserved

PROTON Serial LCD Circuit.



Copyright © Crownhill Associates Limited 2003
All Rights Reserved