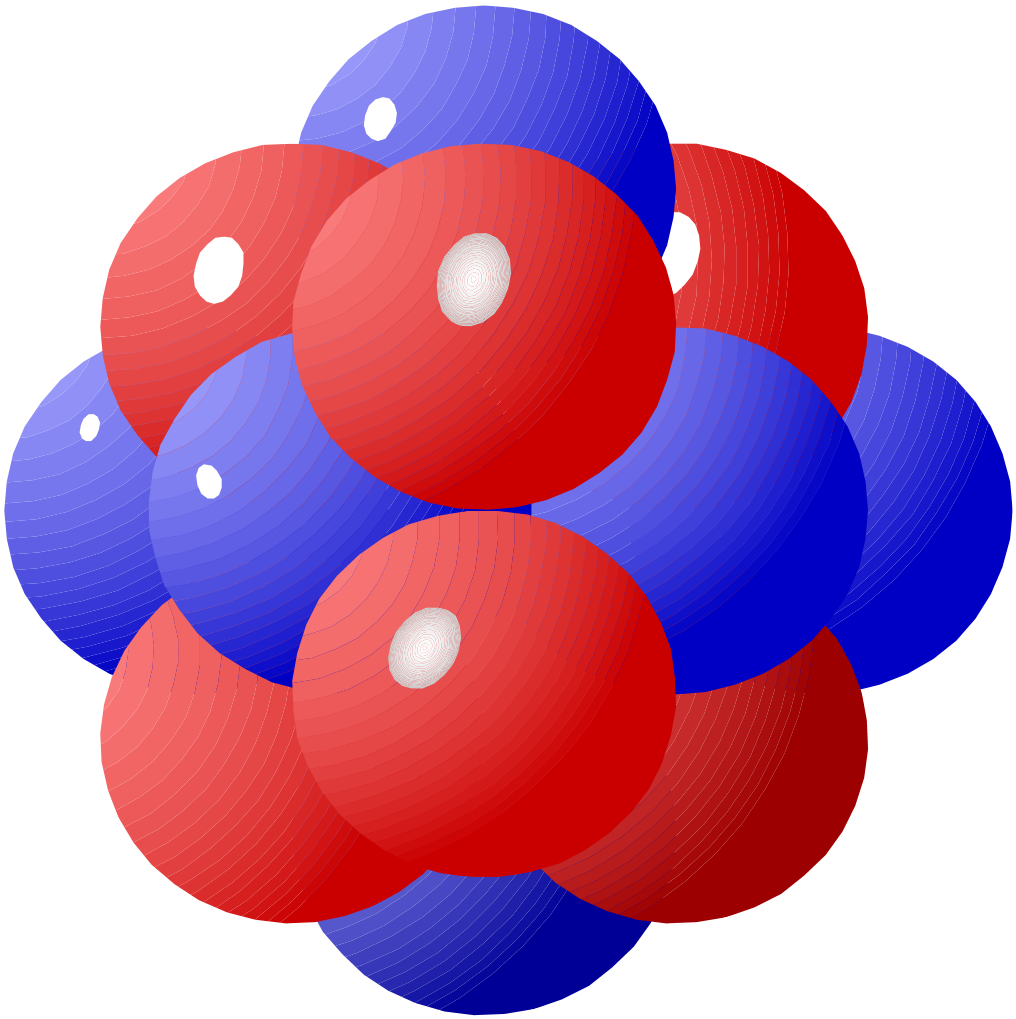
Crownhill

Serial Graphic LCD Interface



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.

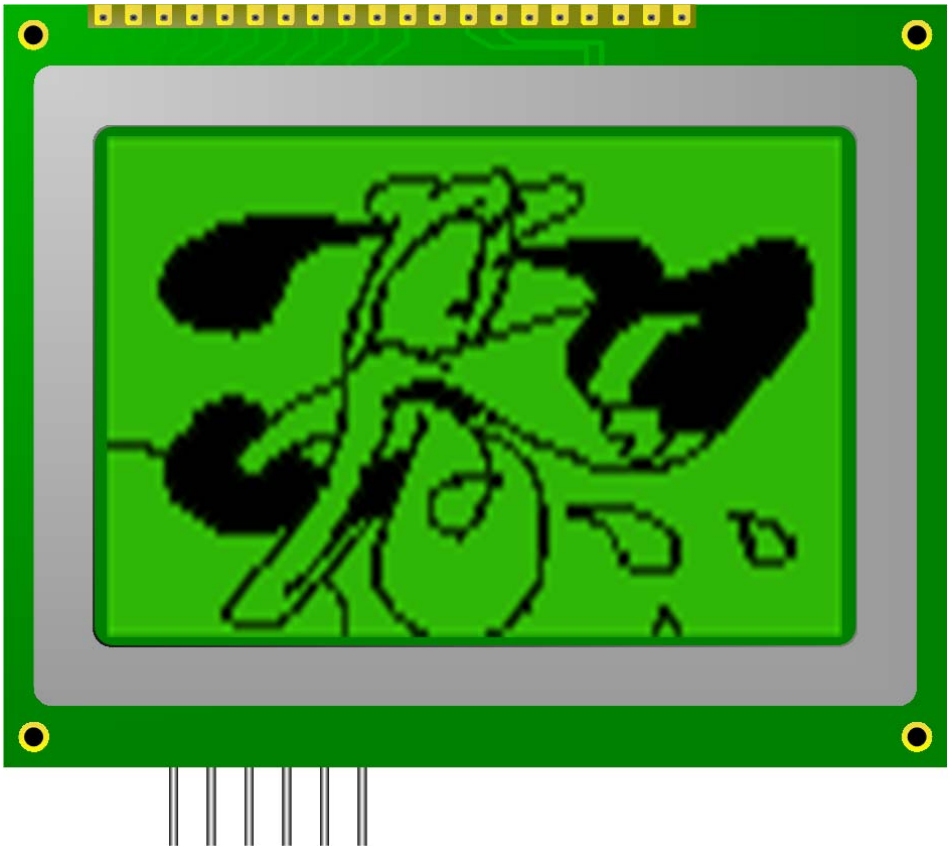
This document was written by Les Johnson, and published by Crownhill associates limited, Cambridge England, 2003

Introduction

Graphic LCDs are not only great fun to use, they also offer a very professional finish to an end product. However, cost has always been a issue, as graphic LCDs are frequently many times more expensive than their alphanumeric counterparts. But thanks to the graphic LCDs that Crownhill supplies, this is now not an issue, as they offer both affordability and flexibility.

This still leaves two chief problems when interfacing to graphic LCDs, board layout and controlling software. Board layout can be a problem because the LCD requires up to 20 pins to be wired. Controlling software can be a problem because of the complex nature of displaying anything of any use on the LCD.

The PROTON+ BASIC compiler takes the hardship out of controlling the LCD, with commands such as PRINT, PLOT, LINE, CIRCLE etc, and now the complex interface is not a problem thanks to the Serial Graphic LCD (shown below).

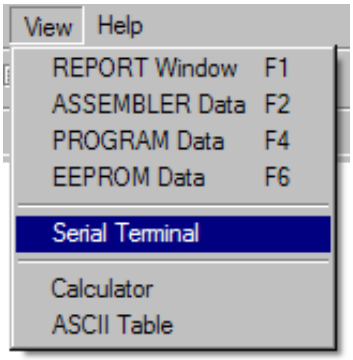


Using a simple RS232 serial interface, you can PRINT, SCROLL, PLOT, SAVE and LOAD screens, (*plus much more*) on the LCD.

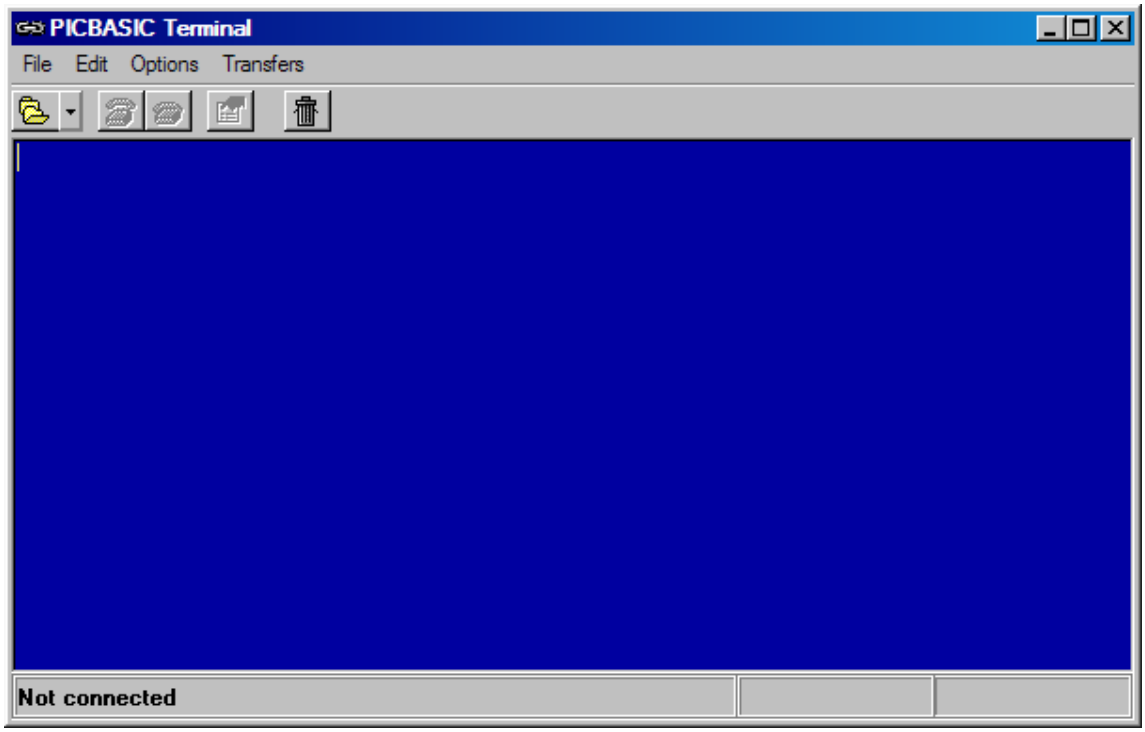
Testing the Serial Interface.

There are two types of interface incorporated within the SGLCD, one for controlling it using the PROTON+ compiler's Serial Terminal (or any serial terminal program) that uses standard ASCII text; which is ideal for testing and demonstrating some of its capabilities. And one that is intended to be controlled from another microcontroller using less cumbersome binary controls (default interface). We'll take a look at the Serial Terminal controlled software to start with, as this is the easiest to demonstrate with.

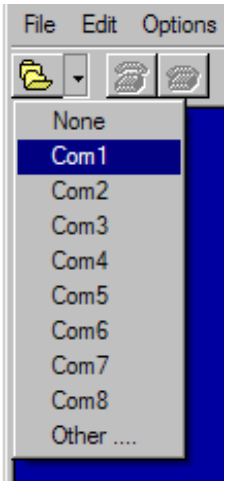
Supply 6 to 9 Volts to the board via its power socket, and a splash screen will be displayed. Open the serial terminal built into the compiler's IDE, by clicking on VIEW->SERIAL TERMINAL (see below).



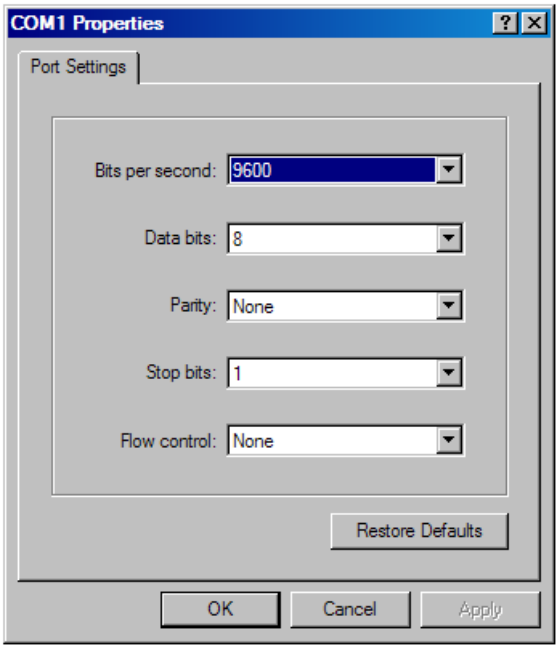
You will be greeted with a window looking something like the screenshot below.



Now the Com port and Baud rate requires setting up. Click on the open Com icon, and a small menu will appear (see below).



Choose the appropriate Com port, according to the setup of your PC. The illustration above shows Com1 being chosen. Note: that the Com port chosen should be the same as the Com port used to download the program to the SGLCD (more details later). When the Com port is chosen, another window will appear that allows the Baud rate to be set (see below).

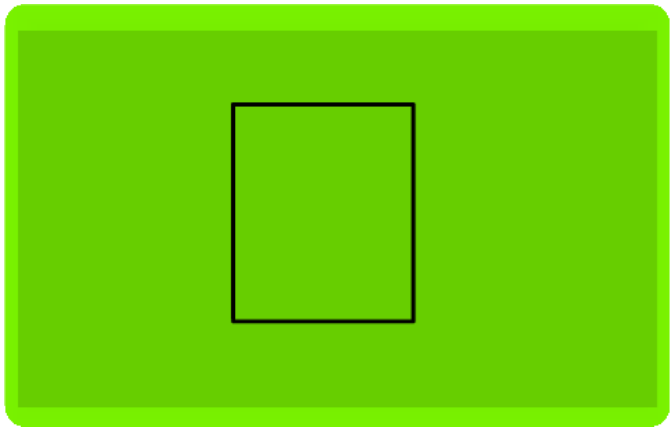


Set the Baud rate to 9600 (as above), and we're ready to send some commands to the graphic LCD.

Just as a starter, type the following text on the terminal all in UPPER CASE and in quick succession.

@A
B3F1F1001

And displayed on the graphic LCD should be a square, as shown below. OK, so it's not an exact square, but that's because of the aspect ratio of the graphic LCD's pixels. They are approx 1.5 times taller than they are wide.



If a square did not appear, then try again but type quicker, and don't forget to type in UPPER CASE.

The collection of characters told the serial LCD to enter the ASCII interface "@A", then draw a square "B" at X position 3F (decimal 63) and Y position 1F (decimal 31), with a RADIUS of 10 (decimal 16), and SET the pixels "01"

Note that all the values are formatted as 2 character HEX values. This is true for all the commands that have parameters, and all future discussions will be based on **Hexadecimal** values unless otherwise stated.

Now type 'B3F1F1000' and the square will disappear because the same XPOS, YPOS and RADIUS have been entered, but now a command to CLEAR the pixels was sent "00"

With that small (but crucial) test carried out, we can continue with a detailed discussion of the commands available.

Graphic LCD Serial Interface Commands

List of commands: -

A… Set or Clear a Single Pixel.
B… Draw a Square.
C… Clear the LCD.
D… Scroll Display Down One Line.
E… Scroll the Display Left a Single Pixel.
F… Scroll the Display Right a Single Pixel.
G… Adjust LCD Backlight Brightness.
H… Change Serial Interface Baud Rate.
I… Rotate the Display Right a Single Pixel.
J… Rotate Part of the Display Right a Single Pixel.
K… Rotate the Display Left a Single Pixel.
L… Draw a Line.
M… Rotate Part of the Display Left a Single Pixel.
N… Scroll the Display Up a Single Pixel.
O… Scroll the Display Down a Single Pixel.
P… Position the Cursor.
Q… Set Response.
R… Draw a Circle.
S… Scroll Display Up One Line.
T… Display Text.
U… Upload a Screen Serially.
V… Rotate the Display Down a Single Pixel.
W… Download a Screen Serially.
X… Load a Screen from Eeprom Memory.
Y… Rotate the Display Up a Single Pixel.
Z… Save a Screen to Eeprom Memory.
@… **@A** Enters the ASCII mode interface.
@M Enters the micro controlled interface.

CHOOSE INTERFACE TYPE

Command: - @

Syntax: - @ A or M depending on interface required

Example: -
@A

The above example, instructed the SGLCD to enter its ASCII interface. A response from the interface informs the user to which interface the LCD is entering.

Entering the text **@M** will place the serial interface into microcontroller mode. Note that this is the default interface of the SGLCD.

DISPLAY TEXT

Command: - **T**

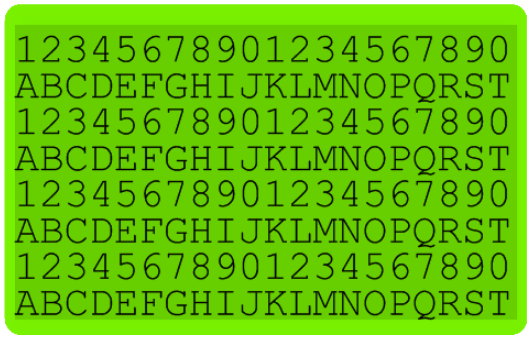
Syntax: - **T** text to display

Example: -
THELLO WORLD

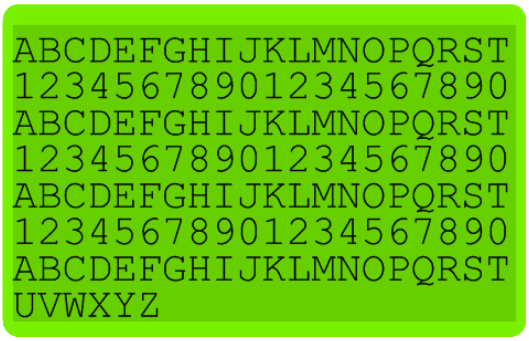
The above example will display the text “HELLO WORLD”. Notice that the text did not appear until typing had finished. This is because the text is stored in a 200 element BYTE ARRAY before being sent to the LCD. However, when typing stops, a TIMEOUT occurs and the text is written to the LCD. This means that up to 200 characters can be entered in one ‘T’ command. Text will be displayed immediately if a NULL (0) is detected in the text stream.

Notes: -
When any text reaches the end of the line, it will move down one and move the X position to 0 and carry on displaying text. Essentially performing a Carriage Return with Line Feed. When the text reaches the end of the display i.e. the end of line on the bottom line (line 07), the text will SCROLL upwards one line, and the top line will disappear.

Try typing ‘T’ followed by many characters until the bottom line is reached. The picture below shows what it could look like.



Now type in a few more characters. i.e. **T**UVWXYZ, and the display will scroll up one line before displaying the new text (shown below).



CLEAR the LCD

Explanation.
Clears the LCD display and positions the cursor at 0,0.

Command: - C

Syntax: -
C

Example: -
B3F1F1001
C

The above example will draw a square (shown earlier) , then typing the single character “C” will clear the LCD.

POSITION the CURSOR

Explanation.
Positions the cursor on the LCD normally prior to a text command being issued.

Command: - P

Syntax: -
P XPOS (*00 to 14*) LINE (*00 to 07*)

Example: -
C
P0003
THELLO WORLD

The above example will clear the LCD, then position the cursor at XPOS 00 and LINE 03 (*remember the LCD lines count from 00 to 07*) before displaying the text “HELLO WORLD”.

SET or CLEAR a SINGLE PIXEL

Explanation.
Sets or Clears a single pixel on the LCD.

Command: - **A**

Syntax: -
A XPOS (*00 to 7F*) YPOS (*00 to 3F*) SET or CLEAR (*00 or 01*)

Example: -
C
A3F2001

The above example will clear the LCD then move to position XPOS 3F and YPOS 20 before setting the pixel near to the centre of the LCD. Altering the last two digits to 00 will clear the pixel: - **A**3F2000

The same pixel has now been cleared.

DRAW a CIRCLE

Explanation.
Draws or Erases a Circle on the LCD at the given X and Y coordinates.

Command: - **R**

Syntax: -
R XPOS (*00 to 7F*) YPOS (*00 to 3F*) RADIUS (*00 to FF*) SET or CLEAR (*00 or 01)*

Example: -
R3F1F1001

The above example will draw a CIRCLE at XPOS 3F and YPOS 1F with a RADIUS of 10. The SET or CLEAR parameter is set to 01, so the pixels will be set. Changing the last parameter to 00 will clear the pixels. For example: - **R**3F1F1000 will clear the circle.

DRAW a SQUARE

Explanation.
Draws or Erases a Square on the LCD at the given X and Y coordinates.

Command: - B

B XPOS (*00 to 7F*) YPOS (*00 to 3F*) RADIUS (*00 to FF*) SET or CLEAR (*00 or 01)*

Example: -
B3F1F1001

The above example will draw a SQUARE at XPOS 3F and YPOS 1F with a RADIUS of 10. The SET or CLEAR parameter is set to 01, so the pixels will be set. Changing the last parameter to 00 will clear the pixels. For example: - **B**3F1F1000 will clear the square.

DRAW a LINE

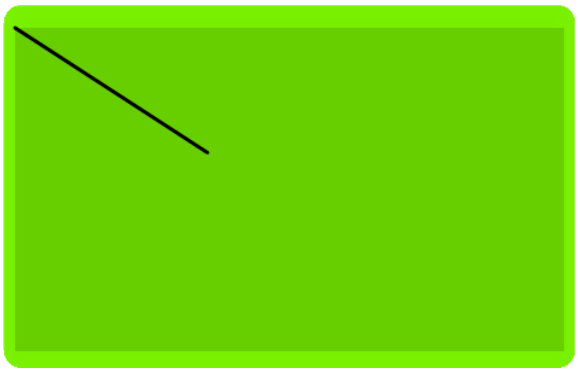
Explanation.
Draws or Erases a Line on the LCD at the given X and Y coordinates.

Command: - L

Syntax: -
L XPOS_START (*00 to 7F*) YPOS_START (*00 to 3F*) XPOS_END (*00 to 7F*) YPOS_END (*00 to 3F*) SET or CLEAR (*00 or 01*)

Example: -
L00002F1F01

The above example will draw a line from XPOS_START 00 and YPOS_START 00 to XPOS_END 2F and YPOS_END 1F.



As with all of the drawing commands, the last 2 HEX digits represent SET or CLEAR the pixel. To erase the line drawn, use: - **L**00002F1F00

SCROLL the DISPLAY UP

Explanation.
Scrolls the LCD up a single line. Any Text or data on the top line will disappear to make room for text or data on the bottom line. The cursor is also set to the beginning of the bottom line.

Command: - S

Syntax: -
S

Example:-
C
P0003
THELLO WORLD
S

The above example will first clear the display (command **C**), then position the cursor at the beginning of line 3 (command **P**) and display the text “HELLO WORLD” (command **T**). The final character, “S”, will SCROLL the display UP one line.

SCROLL the DISPLAY DOWN

Explanation.
Scrolls the LCD down a single line. Any Text or data on the bottom line will disappear to make room for text or data on the top line.

Command: - D

Syntax: -
D

Example:-
C
P0003
THELLO WORLD
D

The SCROLL DOWN example will first clear the display (command **C**), then position the cursor at the beginning of line 3 (command **P**) and display the text “HELLO WORLD” (command **T**). The final character, “D”, will SCROLL the display DOWN one line.

SMOOTH SCROLL the DISPLAY UP

Explanation.
Scrolls the LCD up by a single pixel. Any data on the top row of pixels will disappear to make room for data on the bottom row of pixels.

Command: - N

Syntax: -
N AMOUNT of SCROLLS (*00 to FF*)

Example:-
C
P0005
THELLO WORLD
N3F

The SMOOTH SCROLL UP example will first clear the display (command **C**), then position the cursor at the beginning of line 5 (command **P**) and display the text “HELLO WORLD” (command **T**). It will then smooth scroll the text up the display 3F (63) times.

SMOOTH SCROLL the DISPLAY DOWN

Explanation.
Scrolls the LCD down by a single pixel. Any data on the bottom row of pixels will disappear to make room for data on the top row of pixels.

Command: - O

Syntax: -
O AMOUNT of SCROLLS (*00 to FF*)

Example:-
C
P0002
THELLO WORLD
O3F

The SMOOTH SCROLL UP example will first clear the display (command **C**), then position the cursor at the beginning of line 2 (command **P**) and display the text “HELLO WORLD” (command **T**). It will then smooth scroll the text down the display 3F (63) times.

SMOOTH SCROLL the DISPLAY LEFT

Explanation.
Scrolls the LCD left by a single pixel. Any data on the right column of pixels will disappear to make room for data on the left column of pixels.

Command: - E

Syntax: -
E AMOUNT of SCROLLS (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
E3F

Note. Do not type <CR> this means press return (Carriage Return).

The SMOOTH SCROLL UP example will first clear the display (command **C**), then display some text (command **T**). It will then smooth scroll the text left across the display 3F (63) times.

SMOOTH SCROLL the DISPLAY RIGHT

Explanation.
Scrolls the LCD right by a single pixel. Any data on the left column of pixels will disappear to make room for data on the right column of pixels.

Command: - F

Syntax: -
F AMOUNT of SCROLLS (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
F3F

Note. Do not type <CR> this means press return (Carriage Return).

The SMOOTH SCROLL UP example will first clear the display (command **C**), then display some text (command **T**). It will then smooth scroll the text right 3F (63) times.

SMOOTH ROTATE the DISPLAY UP

Explanation.
Rotates the LCD upwards by a single pixel. Any data on the top row of pixels will appear on the bottom row of pixels.

Command: - Y

Syntax: -
Y AMOUNT of ROTATES (*00 to FF*)

Example:-
C
P0005
THELLO WORLD
Y3F

The SMOOTH ROTATE UP example will first clear the display (command **C**), then position the cursor at the beginning of line 5 (command **P**) and display the text “HELLO WORLD” (command **T**). It will then smoothly rotate the text in an upward direction 3F (63) times.

SMOOTH ROTATE the DISPLAY DOWN

Explanation.
Rotates the LCD downwards by a single pixel. Any data on the bottom row of pixels will re-appear on the top row of pixels.

Command: - V

Syntax: -
V AMOUNT of ROTATES (*00 to FF*)

Example:-
C
P0002
THELLO WORLD
V3F

The SMOOTH ROTATE DOWN example will first clear the display (command **C**), then position the cursor at the beginning of line 2 (command **P**) and display the text “HELLO WORLD” (command **T**). It will then smoothly rotate the text in a downward direction 3F (63) times.

SMOOTH ROTATE the DISPLAY LEFT

Explanation.
Rotates the LCD anticlockwise by a single pixel. Any data on the left column of pixels will re-appear on the right column of pixels.

Command: - K

Syntax: -
K AMOUNT of ROTATES (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
K7F

Note. Do not type <CR> this means press return (Carriage Return).

The SMOOTH ROTATE LEFT example will first clear the display (command **C**), then display some text (command **T**). It will then smooth rotate the display in an anticlockwise direction 80 (128) times. And the text will end where it started.

SMOOTH ROTATE the DISPLAY RIGHT

Explanation.
Rotates the LCD clockwise by a single pixel. Any data on the right column of pixels will re-appear on the left column of pixels.

Command: - I

Syntax: -
I AMOUNT of ROTATES (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
I80

Note. Do not type <CR> this means press return (Carriage Return).

The SMOOTH ROTATE RIGHT example will first clear the display (command **C**), then display some text (command **T**). It will then smooth rotate the display in a clockwise direction 80 (128) times. And the text will end where it started.

SMOOTH ROTATE PART of the DISPLAY LEFT

Explanation.
Rotates a user defined section of the LCD anticlockwise by a single pixel. Any data on the left column of pixels will re-appear on the right column of pixels.

Command: - M

Syntax: -
M XPOS START (*00 to 7F*) YPOS START (*00 to 3F*) XPOS END (*00 to 7F*) YPOS END (*00 to 3F*) AMOUNT of ROTATES (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
M05003B0237

Note. Do not type <CR> this means press return (Carriage Return).

The above example will first clear the display (command **C**), then display some text (command **T**). It will then smooth rotate part of the display in an anticlockwise direction 37 (decimal 55) times. And the text will end where it started.

SMOOTH ROTATE PART of the DISPLAY RIGHT

Explanation.
Rotates a user defined section of the LCD clockwise by a single pixel. Any data on the right column of pixels will re-appear on the left column of pixels.

Command: - J

Syntax: -
J XPOS START (*00 to 7F*) YPOS START (*00 to 3F*) XPOS END (*00 to 7F*) YPOS END (*00 to 3F*) AMOUNT of ROTATES (*00 to FF*)

Example:-
C
THELLO WORLD <CR>
HOW ARE YOU
J05003B0237

Note. Do not type <CR> this means press return (Carriage Return).
The above example will first clear the display (command **C**), then display some text (command **T**). It will then smooth rotate part of the display in a clockwise direction 37 (decimal 55) times. And the text will end where it started.

SAVE a SCREEN to EEPROM MEMORY

Explanation.

Save the current LCD display to a particular portion of the I^2C eeprom memory. Within the on-board serial eeprom, which contains 32 Kbytes of memory, thirty two screens can be saved. As you've probably gathered, a screen consists of 1 Kbytes (1024 bytes).

Because of the delay required for writing to the serial eeprom, a screen save may take a few seconds. The SGLCD board's serial interface will acknowledge when a save is complete.

Command: - Z

Syntax: -

Z LOCATION in MEMORY (*00 to 1F*)

Example: -

C

THELLO WORLD

Z10

In the example above, the LCD is first cleared (command C), and text is displayed (command T). Once the text has appeared on the LCD, the SAVE SCREEN command is issued (Z) with the location to save the screen. While the screen is being saved to eeprom memory, the SGLCD will not respond to any further commands. After a few seconds, an acknowledge character "A" will be transmitted from the SGLCD to signal the screen has been saved.

LOAD a SCREEN from EEPROM MEMORY

Explanation.

Load a screen (saved earlier) from a particular portion of the I^2C eeprom memory. Within the on-board serial eeprom,, which contains 32 Kbytes of memory, Thirty two screens can be stored. As you've probably guessed, a screen consists of 1 Kbytes (1024 bytes).

Command: - X

Syntax: -
X LOCATION in MEMORY (*00 to 1F*)

Example: -
C
X10

The example above should be used after the SAVE to SCREEN example has been carried out. In the example, the LCD is first cleared (command C), and the LOAD SCREEN command is issued (X) with the location to load the screen from. The previously saved screen will then be loaded on to the LCD.

Example2: -
C
X01

The SGLCD's eeprom is preloaded with several sample screens. Example2 will load the screen stored at location 01 and the display shown below should appear.

UPLOAD a SCREEN

Explanation.

Most complex images displayed on the LCD are usually created using the PC, so a means of uploading the images to the LCD is required. That's what the UPLOAD SCREEN command is for. It accepts a list of HEX data; where each HEX byte represents a byte on the LCD working from top left of the LCD to bottom right. Once the image is displayed on the LCD, the SAVE SCREEN command can store it in eeprom memory.

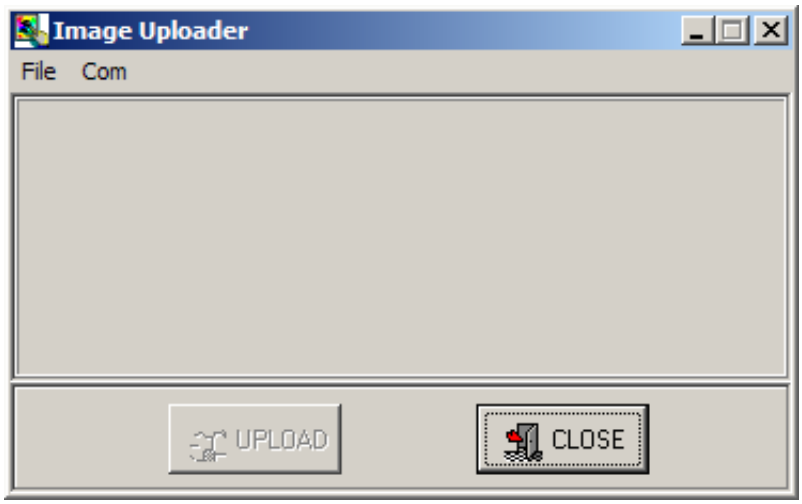
Command: - U

Syntax: -
U

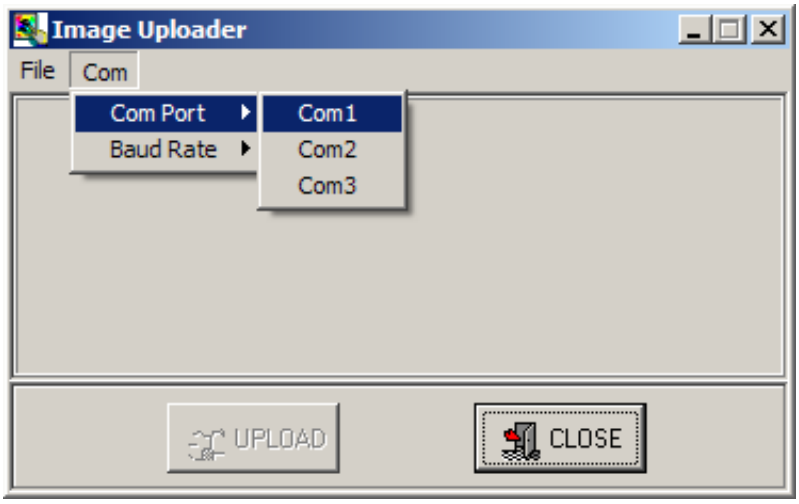
Example: -

In order to demonstrate uploading an image to the LCD, a small executable program has been created that will load an image from the PC, and upload the image to the LCD via the serial port.

Run the program **Image_Upload.exe** (downloadable from www.picbasic.org).

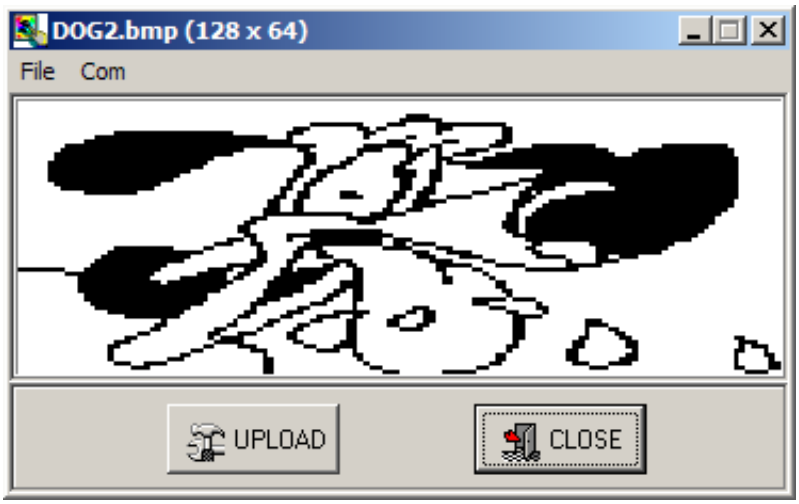


Choose the com port that the SGLD is attached to (see overleaf).



In the screen shot shown above, com port 1 was chosen. Leave the baud rate as it is for now, because it defaults to 9600, which is the default baud rate of the SGLCD. Once the com port is chosen, the UPLOAD button will be enabled.

Click on FILE, and load one of the pictures located in the same folder as the executable. The screen shot below shows the picture DOG1.BMP.



Ensure power is applied to the SGLCD, and that the splash screen has completed its journey to the bottom of the display, then press the UPLOAD button.

An image will be produced on the SGLCD's display, scanning from top left to bottom right. Once this has completed, close the IMAGE UPLOADER, and save the screen to eeprom memory using the SAVE SCREEN command. Remember where you placed it, as we'll use this in the next command's example.

DOWNLOAD a SCREEN

Explanation.

Images produced on the LCD's display can be saved to eeprom memory for long term storage, or downloaded back to the PC for inclusion into other programs. The DOWNLOAD IMAGE command, sends the data from the display in the format of **CDATA** statements, ready to be copied and pasted into another PROTON+ BASIC program.

Command: - W

Syntax: -
W

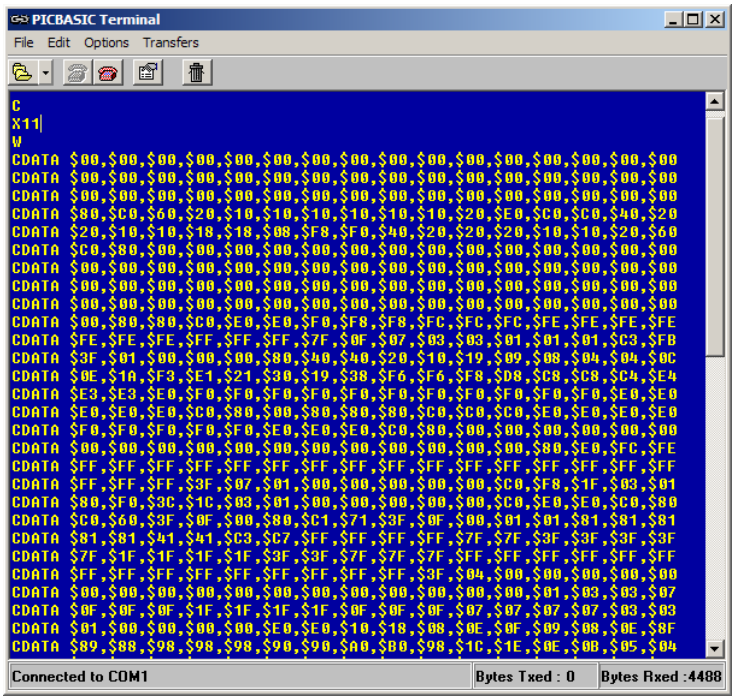
Example: -

In order to demonstrate downloading an image to the PC, you must first have created or loaded one. For this example, we'll use the previously uploaded image of the dog (see UPLOAD a SCREEN).

Open the Serial Terminal and enter the following commands: -

C
X11
W

The above commands clear the LCD (command C), and load the previously saved image from location 11. The W command then transfers the image to the Serial terminal (shown below).



LCD BACKLIGHT CONTROL

Explanation.

The graphic LCD supplied with the SGLCD has a backlight facility, but the default for this is OFF in order to conserve power. The brightness of the backlight can be controlled using the BACKLIGHT CONTROL command, and uses the microcontroller's hardware PWM (Pulse Width Modulation) feature to further save power usage when the backlight is in use.

Command: - G

Syntax: -
G BRIGHTNESS LEVEL (*7F to FF*)

Example: -
GFF

The above example will illuminate the LCD's backlight to full brightness level.

Example 2: -
G00

Example 2 will extinguish the LCD's backlight.

Example 3: -
GF0

Example 3 will dimly illuminate the LCD's backlight.

Generally, a value between E0 to FF is all that's required for the full illumination swing of the backlight.

ENABLE/DISABLE RESPONSE

Explanation.

Each command available in the SGLCD takes a finite amount of time to complete, and while this is unimportant when using the serial terminal as an interface, it is not the case if a program is automatically using the interface (such as the image uploader). What is then required is a method of knowing that the SGLCD has finished its current task and is ready for the next. This is the job of the ENABLE/DISABLE RESPONSE command. When enabled, each command will send an acknowledgement of characters <CR> 'O' <CR> if all was OK, and other ACKS if something went wrong will performing the specific command.

Note. <CR> represents Carriage Return which has a value of 13.

Command: - Q

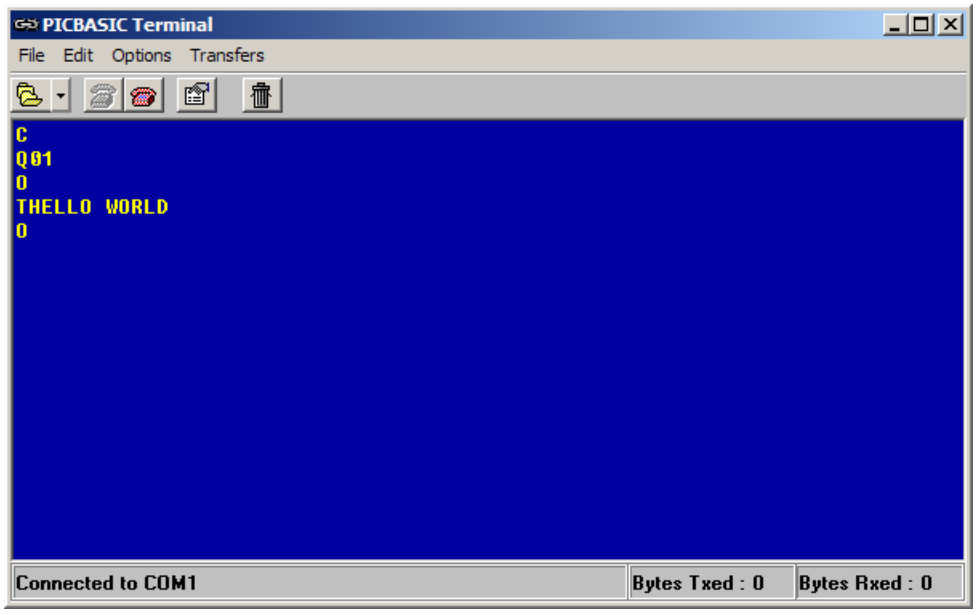
Syntax: -

Q DISABLE or ENABLE RESPONSED (*00 or 01*)

Example: -

C
Q01
THELLO WORLD

The example first clears the LCD (command C), then enables RESPONSES (command Q). The SGLCD will immediately respond with the ack character 'O'. Text is then displayed (command T) and the SGLCD responds with the ack character 'O' when the text has finished being displayed. You should see the same results as shown in the screen shot below.



To disable the responses, then issue the command Q00. The SGLCD will give a response of 'O' to signify OK.

Other responses are: -

N… Command not recognised. This will also be produced if the RETURN key is pressed, as the serial interface does not recognise this as a command.
T… Timeout occurred. If the information is not entered in a reasonable time period, a timeout will occur to allow the serial interface to look for another command.

ADJUST BAUD RATE

Explanation.
By default, the SGLCD's serial interface works at a speed of 9600 Bits Per Second (Baud). However, it is capable of working at much slower or higher baud rates. Up to 115200 Baud in fact.

Command: - H

Syntax: -
H BAUD RATE VALUE (Decimal value 150 to 115200) <CR>

Example: -
C
THELLO WORLD
H19200 <CR>

< *Adjust baud rate of serial terminal to 19200 before typing the following text* >

THELLO AGAIN

The example above will first clear the LCD (command C), then display some text at the current baud rate (9600 by default). The baud rate will then be changed to 19200 by the H command. A carriage return must be typed after the decimal baud rate is entered. The SGLCD will then respond with an ack of 'O'.

The baud rate of the SGLCD is now set to 19200, so change the baud rate of the Serial Terminal, and type in the second text message.

The new baud rate is not retained by the SGLCD after power down, therefore baud rate negotiation must be accomplished upon use.

Microcontroller Interface Firmware

The SGLCD will most often be accessed by another microcontroller, therefore HEX parameter values can become cumbersome to use. This is the reason for the default 'microcontroller interface', the command values stay the same, but the parameters are not required to be in ASCII HEX digits.

For example to draw a circle on the LCD using the PROTON+ compiler, the **RSOUT** command could be used.

```
RSOUT "R" , 63 , 32 , 10 , 1
```

Notice how the actual command is still an ASCII 'R', but the parameters are now standard decimal values.

Negotiating a Response.

One thing that is profoundly different is the way the SGLCD responds to a command.

Each command takes a predetermined time to performs its task, and the controlling microcontroller needs to know when the command is finished and able to accept another.

The SGLCD always send an acknowledge after a command, but does so with a minimal delay of 1ms. This may be too quick for the controlling microcontroller to re-adjust itself from transmitting the command to accepting the ack from the serial port, so a method of calibrating the ack response is always a wise precaution if the controlling microcontroller is using a clock speed of less that 8MHz. Code that could be used to calibrate the ack response is listed below.

```
Dim RESPONSE_TIME as Byte
Dim BLANK as Byte
RESPONSE_TIME = 0          ' Set initial response to 0
Repeat                     ' Create a loop
 NEGOTIATE_RESPONSE:       ' Timeout label
 Inc RESPONSE_TIME         ' Increment the RESPONSE time every cycle
 HRSOUT "Q",RESPONSE_TIME      ' Send RESPONSE delay
' Exit loop if response received
Until HRSIN ,{100,NEGOTIATE_RESPONSE} = "O"
```

The code above starts at the smallest response time which is 1, then sends the RESPONSE command of 'Q', and waits 100ms for a response of character 'O'.

If a response was not received (signified by a timeout) , then the response time is incremented until a response of character 'O' is received correctly.

Once the response time its established, the SGLCD will continue using the time delay until power is removed, or another response calibration is carried out.

In tests, a response time of 2ms was found to be fairly standard using a controller operating with a 4MHz crystal.

The syntax for the RESPONSE command is: -

Q DELAY BETWEEN RESPONSES (0 to 255)

Once the response time has been negotiated and calibrated, it's a simple process for sending commands: -

Send Command
Wait for acknowledge

Which relates to: -

```
RSOUT "R" , 63 , 32 , 10 , 1   ' Draw a circle
BLANK = RSIN                   ' Wait for acknowledge
```

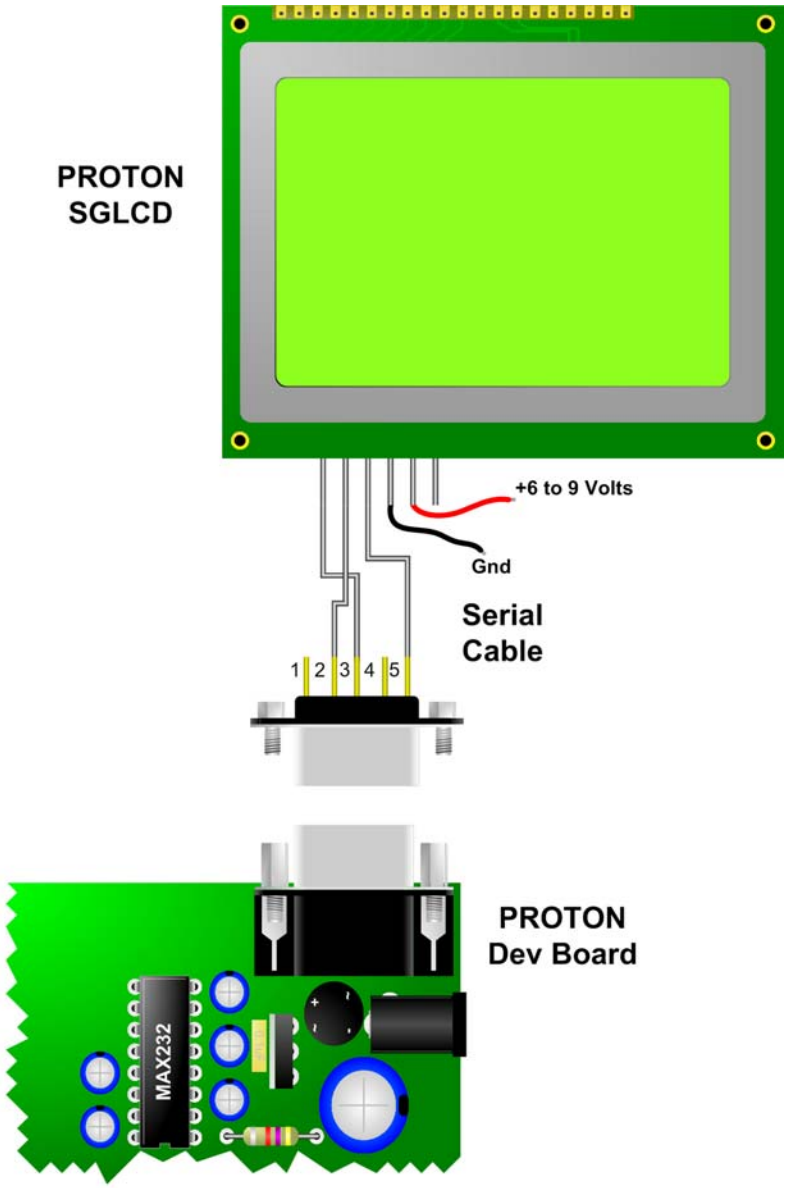
The **RSIN** command used for receiving the ack from the serial interface could also have a timeout value to prevent any lock-ups if the ack byte is missed. Of course, not only **RSIN/RSOUT** can be used. Any of the compiler's serial commands can be used to access the SGLCD's serial interface.

DOWNLOADING a SCREEN

Another difference with the microcontroller firmware is the data received by the DOWNLOAD SCREEN command. Instead of the data being formatted for **CDATA** statements, it is simply transmitted as a series of values.

Shown overleaf is a demonstration program for use with the PROTON Development board, controlling the SGLCD. Each device is controlled via its serial socket.

The illustration below shows the connections of the serial cable and how it attaches the SGLCD to the PROTON Development board.



Shown overleaf is the demonstration program mentioned earlier. The program animates the 5 sample images stored in the eeprom when it's shipped.

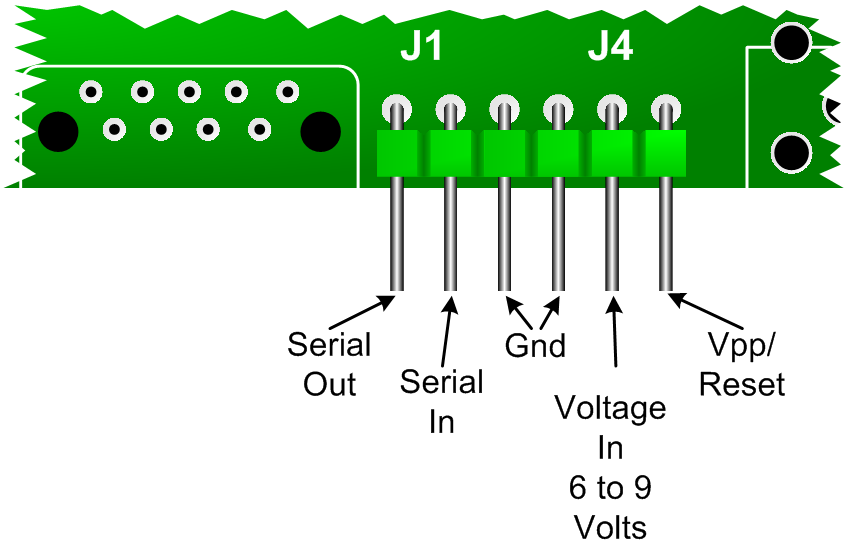
I hope you agree that even with only 5 images, the effect is rather impressive.

PROTON+ Compiler Demonstration

```
' Interface test for microcontroller firmware version
' of SGLCD
'
' For use on the PROTON PICmicro Development Board
'
' Displays the 5 demo images stored in serial eeprom

Include "PROTON_4.INC"          ' Use the PROTON Dev board at 4MHz
Dim BLANK as Byte
Dim RESPONSE_TIME as Byte
Dim DISPLAY_LOOP as Byte
Delayms 500                     ' Wait for PICmicro to stabilise

' Calibrate the response time
RESPONSE_TIME = 0               ' Set initial response to 0
Repeat                          ' Create a loop
NEGOTIATE_RESPONSE:             ' Timeout label
Inc RESPONSE_TIME         ' Increment the RESPONSE time every cycle
HRSOUT "Q",RESPONSE_TIME            ' Send RESPONSE delay
' Exit loop if response received
Until HRSIN ,{100,NEGOTIATE_RESPONSE} = "O"

HRSOUT "G" , 255                ' Backlight to full
BLANK = HRSIN                   ' Wait for a response

' Display the moving pictures
While 1 = 1                     ' Create an infinite loop
DISPLAY_LOOP = 1                ' Clear the display loop counter
Repeat                          ' Form a loop for the frames
HSEROUT ["X", DISPLAY_LOOP]     ' Load an image from eeprom memory
BLANK = HRSIN                   ' Wait for a response
Inc DISPLAY_LOOP                ' Advance a frame
Until DISPLAY_LOOP > 5          ' Exit after 5 frames
Wend                            ' Do it forever
```

Connections to the SGLCD.

The SIL header pins between the serial and power sockets, are for connections to the board using a ribbon cable. The pinouts for the header are shown below.



SERIAL IN connects to pin 13 (R1 In) of the MAX232 RS232 transceiver, and carries serial data to the on-board PICmicro.

SERIAL OUT connects to pin 14 (T1 Out) of the MAX232 RS232 transceiver, and carries serial data from the on-board PICmicro.

GND is the common ground (0v) connection.

VOLTAGE IN connects directly to the input of the 7805 voltage regulator, and can be any DC voltage from 6 to 9 Volts.

VPP/RESET connects to the on-board PICmicro's MCLR pin, and can be used to reset the SGLCD.