FGC

GLIC-K

Graphic LCD Interpreter Chip

The GLIC-K1 ic is an interpreter between a PICAXE or Basic Stamp and a Graphical LCD.

The GLIC-K1 is designed to work with a 128 x 64 Samsung **KS0108** based GLCD.

Note: There are other controllers compatible with the KS0108 which may function OK, but we would recommend only using GLCDs recommended by FGC - otherwise certain 'funnies' may occur which cannot be retrofixed.

FOR Toshiba T6963 based 128 x 64 Graphic LCDs (GLCDs) use FGC's GLIC-T chip.

Schematic.

In a similar way to serial LCD driver ICs, the GLIC-K1 is connected between your PICAXE / Basic Stamp and a KS0108 controlled GLCD.

Similarly, GLIC-K1 receives serial commands and data from your PICAXE and carries out the instructions and controls the GLCD.

This makes GLCD control much easier for the designer and reduces code space required - important with PICAXE.

GLIC-K1 introduces some novel and useful functions which require the EEPROM and if necessary, the GLICTalk-K PC software.

Whilst the EEPROM is optional, several of these functions will NOT work without it.

We would highly recommend that you include the EEPROM into your initial design.
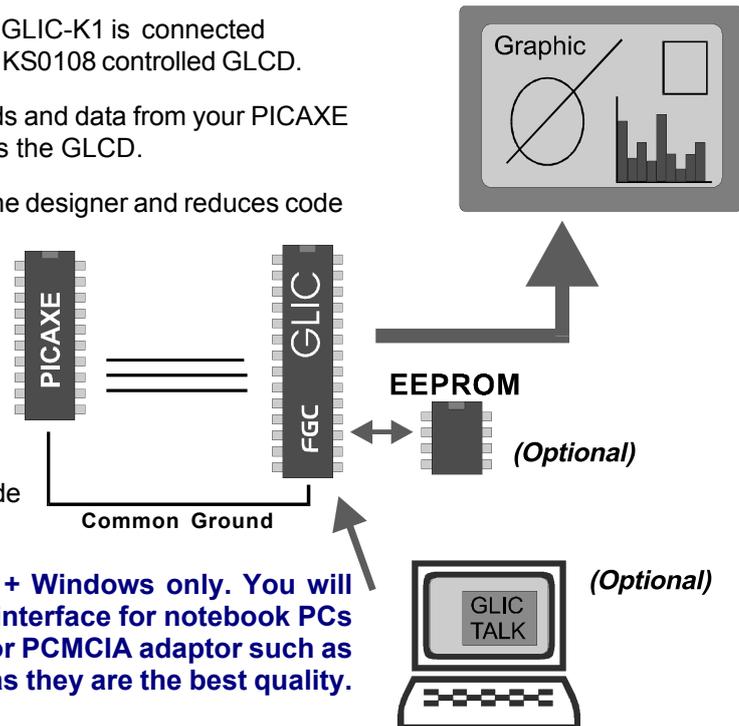
GLICTalk-K is a software app for PC + Windows only. You will require a serial port. If you require an interface for notebook PCs we recommend a dedicated interface or PCMCIA adaptor such as the models made by Brain Boxes Ltd as they are the best quality.

Graphic

PICAXE

GLIC

FGC

EEPROM

(Optional)

Common Ground

GLIC TALK

(Optional)

UPDATE 1: COMMAND CODES
PRINT AT - **MODIFIED**
FILLED SQUARE - **NEW**

This manual is split into 3 sections.

1. **The Circuit Diagram of the FGC Interface.**

2. **Summary of Command Codes.**

3. **Detailed description of Command Codes with examples.**

PICAXE

BASIC STAMP

General Notes Before You Start.

1. GLIC-K1 requires a 5 Volt well regulated power supply. It will work down to 4.5 Volts DC but it is recommended that a good 5 V supply is used. It can share a 5 volt supply with your PICAXE device, but we recommend a 220nF decoupling capacitor across the supply pins of PICAXE & GLIC-K1.

2. GLCDs require a 5V power supply too. Most GLCDs will have a backlight option and may require up to 200mA current supply. It is wise to have a decoupling capacitor at the GLCD supply pins too.

3. GLCD Contrast. GLCDs require a Negative -V supply (see Circuit Diagram) to make the display visible.
DO NOT confuse *negative* with *ground (0V)*. **DO NOT** apply Negative power without the main 5V supply.
The GLCDs on our 'recommended list' have built-in voltage inverters.

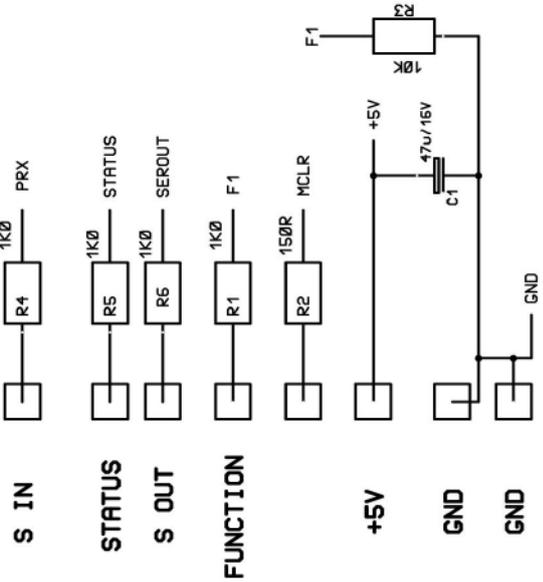# Minimum Circuit Diagram including EEPROM (24LC512)

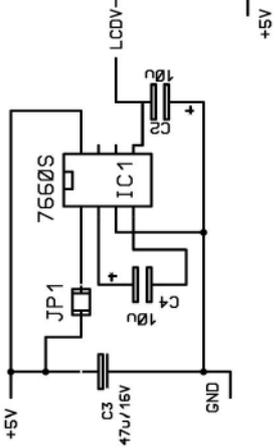**Copyright FGC 2006**

R14 Limits the current through the GLCD backlight. The maximum permitted depends on the make of GLCD. Check GLCD Data Sheet.
Guide:
40mA try 100R
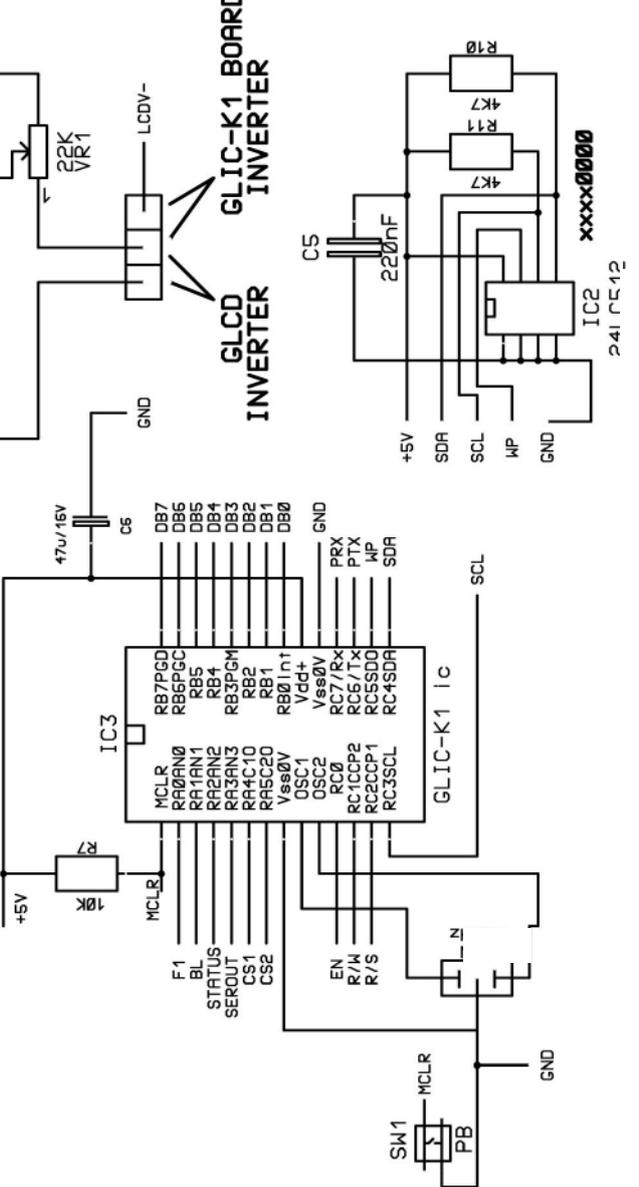150mA try 20R   (Check GLCD Data)
SEE ADDENDUM A3

GLCD Backlight Current Limiter
SEE NOTES!

BOARD INVERTER

Serial TxRx to PC RS232
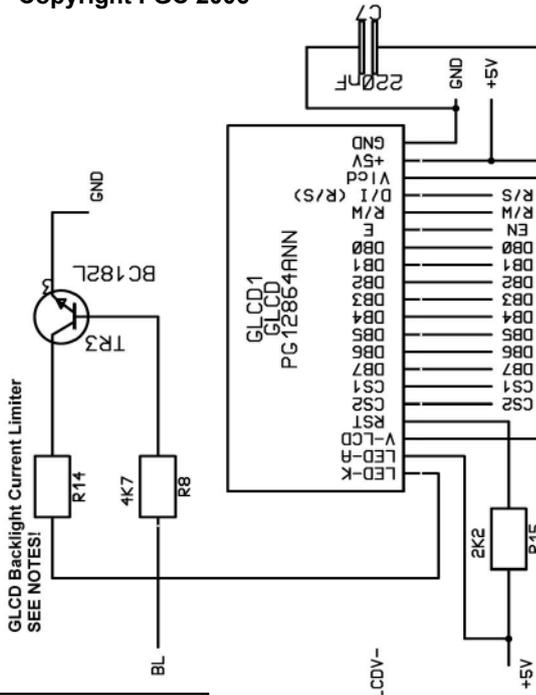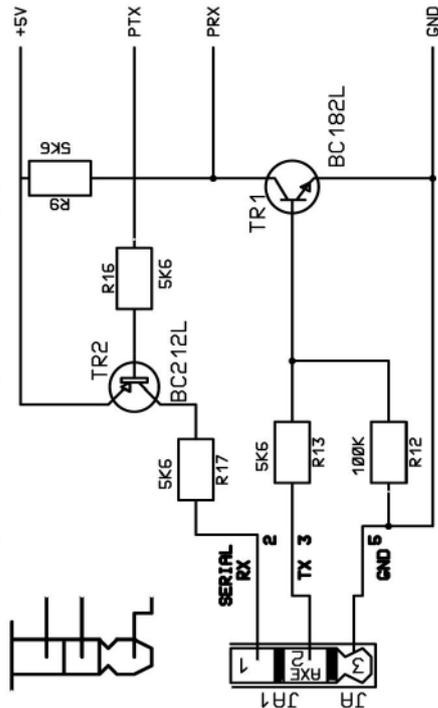
If you set 'Function' high during power-up the version code is displayed on the GLCD.

Circuit Diagram Overview.

This circuit diagram was constructed using the 'signal' method for indicating track connections.

This simply means that many physical connection lines are replaced by symbols on the diagram, such that for example, if you see a point marked 'GND' it connects to any other point marked 'GND'.

For example, the GLIC-K1 IC; Pin 8 (Vcc/0V) has a short line with 'GND' next to it. This means it connects to Ground and any other point with 'GND' e.g. GLIC-K1 IC Pin 19 which is also a ground.

You will see the 'signal' connections between GLIC-K1 and the GLCD too.

This may take a little getting used to, but prevents the drawing being covered in lines which can, on occasion, produce cross-eyes in the circuit designer.

GLIC Pinout Description.

The GLIC is a pre-programmed PIC Microcontroller 18F2520.

It is not a PICAXE. The code is compiled and it has no bootloader.

For the *electrical* specifications of the i.c. consult the data sheet from Microchip Technology Inc.
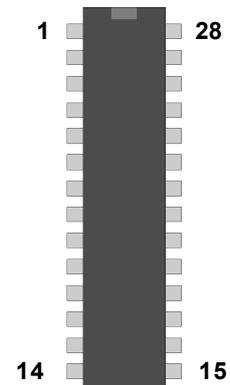
Pin Number	Signal Name	Description
1	MCLR	Reset by taking to ground
2	F1	Low= Normal use High=Display Command
3	BL	Connect to Backlight Transistor
4	STATUS	High=Busy Low = Ready to receive.
5	SEROUT	Optional return to PICAXE / Stamp
6	CS1	**GLCDConnection**
7	CS2	**GLCD Connection**
8	GND	**Circuit GROUND (0V)**
9	Osc1	Connect to res/crystal 20MHz
10	Osc2	Connect to res/crystal
11	EN	**GLCD Connection**
12	R/W	**GLCD Connection**
13	R/S	**GLCD Connection**
14	SCL	**EEPROM SCL**
15	SDA	**EEPROM SDA**
16	WP	**EEPROM WP**
17	PTX	PC Interface section
18	PRX	PC Interface section & **Serial Command IN**
19	GND	**Circuit GROUND (0V)**
20	Vdd	**+5V Regulated supply**
21	DB0	**GLCD Connection**
22	DB1	
23	DB2	
24	DB3	
25	DB4	
26	DB5	
27	DB6	
28	DB7	
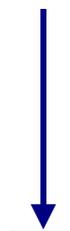
PICAXE to GLIC-K1 Connections

You must have a common Ground (0V) and a PICAXE Output pin for Serout commands to GLIC IC pin number 18 (PRX). For Stamp or PICs, set a convenient pin to 'output'.
The GLIC 'Status' and 'Serout' pins are optional. We'll look at this again during the Code section.
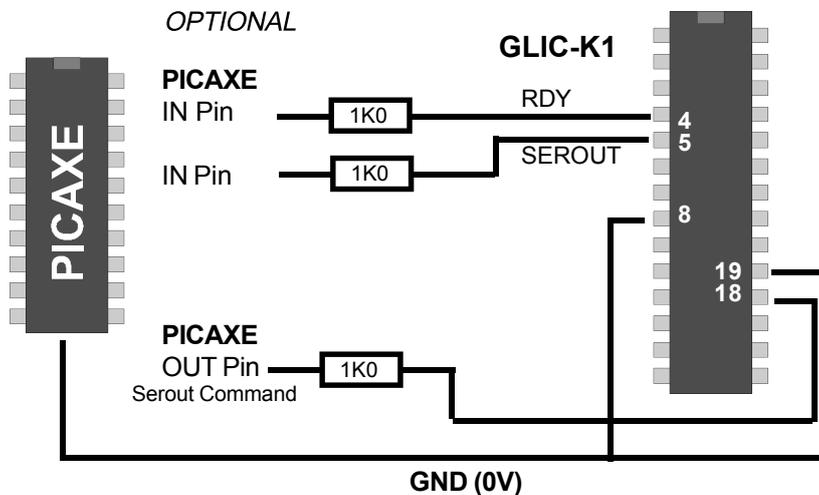
Optional PICAXE to GLIC-K1 Connections.

The circuit requires you to connect PICAXE Serout pin to GLIC 'PRX' via a 1K0 resistor.

There are two other optional connections; **STATUS & SEROUT**

.

STATUS   is held HIGH whilst GLIC is processing and LOW when ready to receive PICAXE data. If you don't use this you MUST include pauses between PICAXE commands.

SEROUT   (in conjunction with **F1** pin held high) bounces any invalid command number back to your PICAXE so that you can analyse any comms errors in the Debug window.

NOTE: If you are using separate 5V supplies to PICAXE/Stamp and GLIC-K1 in your own design then you **MUST** include the resistors as shown. If you are using the FGC interface then they are included already.

Why make the 'STATUS' an Option?

The GLIC-K1 IC receives commands in a T2400 serial format from your PICAXE. It doesn't require an end-of-message character but works on timing.

So, during operation, the GLIC waits .... listening for any data from your PICAXE.

When GLIC detects data , it puts the data into a temporary store called a buffer. Because it doesn't know when your PICAXE is going to stop transmitting it waits to see if any more data is on its way.

If it detects a suitable break (30mS), it assumes PICAXE has stopped 'talking'.

It then gets on with the processing and controlling the GLCD. Depending on the command this may take from 85mS to nearly 400mS. During this time it holds the STATUS pin High (~5V). If any data came during this time it would upset the next command.

So, either you use the 'STATUS' pin to check that GLIC-K1 is ready (STATUS High), or, you have to add short pauses between commands.

The 'short pause' could be PICAXE 'Pause' command, **or**, your programme may be doing some other task such as calculation or controlling another device, thus in effect 'parallel' processing.

The 'Status' pin can be checked at any time to ensure commands do not overlap.

PC Interface. (*Optional) (Standard issue on FGC GLIC-K1 Interface board*)

As you know, the PICAXE requires a simple interface to connect to your PC using the serial cable.

The GLIC-K1 also needs an interface. Normally this would require an inverter and level shifter circuit such as those based on the MAX232 or similar.

However, our interface is much simpler and uses very little power. As a result it does not conform to true RS232 levels and may not work with some 'antique design' computers.

Note: The transistor values shown are a guide and any similar spec devices will work.
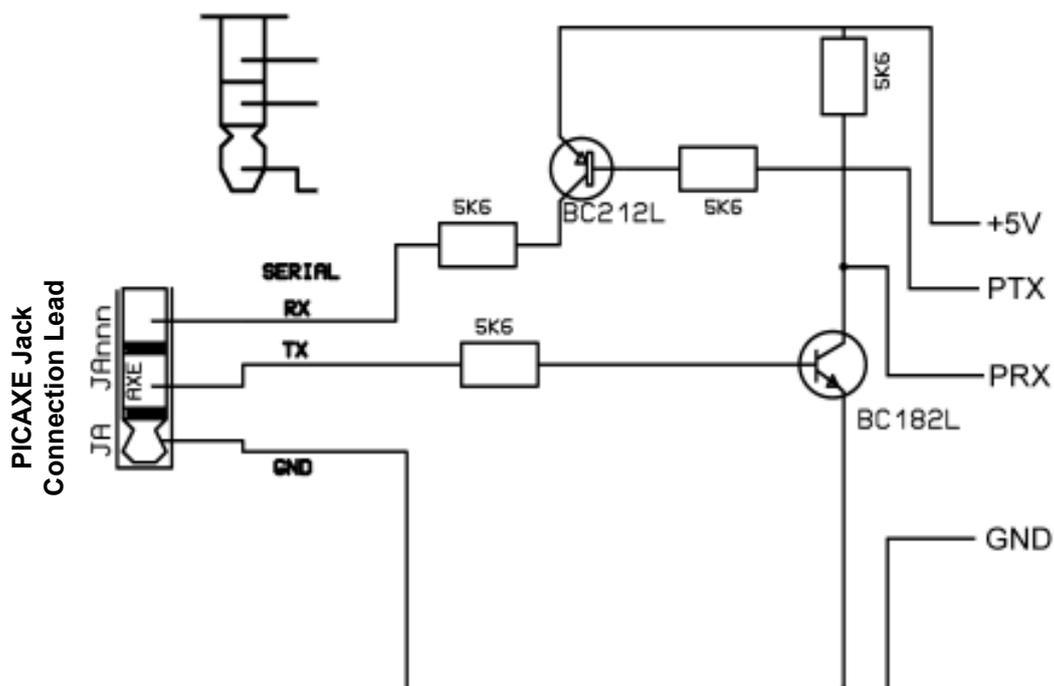
Again, this is in 'signal' format.

+5	connects to your 5 Volt supply
PTX	connects to GLIC IC pin 17
PRX	connects to GLIC IC pin 16 - **AND** also to the PICAXE / Stamp Serout line.
GND	connects to the common Ground 0V - NOT negative.

Please note.
As mentioned on the previous page , your PICAXE 'Serout' pin is also connected to 'PRX' in the above PC interface. It is IMPORTANT that PICAXE does NOT send data at the same time as the PC. So never connect the PC whilst your PICAXE is running and sending commands to the GLIC. Ideally, connect either and NOT both. It won't explode it just corrupts data.

Negative Power Supply.

Unlike most standard character LCDs, GLCDs require a **negative** voltage to make the pixels visible. Many GLCDs have a built-in inverter. You can jumper-select this option on our interface.

For those GLCDs that do not inlcude an inverter, you must supply a negative voltage. The FGC GLIC-K1 interface has an ic slot to insert an inverter chip. You will then have to jumper-select the option to make the GLCD pixels visible.

Contrast is controlled by a 20K potentiometer. - see page 2 Circuit for details.

Resistor R14
Please read the appropriate GLCD Data Sheet to calculate the value.

Command Format:

Serout   <PICAXE Pin> , T2400 , (Command Number, Data........)

Command Summary.

Text Functions:	C/Num	Description:
PrintAT	*100*	Displays your text (up to 20 chars) at Column X, Row Y
Store Text	*50*	Stores text blocks to EEPROM location. Up to 100 blocks.
Get Text & Print AT	*101*	Retrieve selected text from EEPROM and Print AT X, Y.
Graph X Axis Title	*184*	Print X-axis title after bar chart plot (max 14 chars)
Graph Y Axis Title	*185*	Print Y-axis title after bar chart plot (Max 6 chars)
Graph Heading	*186*	Print Bar Chart Heading.
Print Page	*123*	Prints text from EEPROM as a page. Handy for info/help.
Clear Screen	*10*	Clear both Text and Graphics screen

Graphical:

Plot	*150*	Plot individual pixel point at X, Y. Multi-command.
UnPlot	*151*	Erase pixel point at X, Y.
Draw Line	*155,1*	Draw a line between X1, Y1 and X2, Y2. Multi-command.
UnDraw Line	*155,0*	Erase a line specified.
Draw Circle(s)	*160,1*	Draw a Circle, Centre X, Y with radius R. Multi-command. Erase
UnDraw Circle(s)	*160,0*	Circle specified.
Draw Square(s)	*165,1*	Draw a Square, centre X, Y with edge E. Multi-command. Erase
UnDraw Lines(s)	*165,0*	Square specified.
BarChart	*180*	Plot a Bar Chart with up to 30 (1 byte) data points. (Function automatically calculates height and bar width)
Draw Pointer 1	**200**	Draw pointer1; Centre X, Y, Length L, Angle , SetClear
Draw Pointer 2	**210**	Draw pointer2; Centre X, Y, Length L, Angle , SetClear
Get Bitmap and Display	**221**	Retrieve a bitmap image from 24LC512 EEPROM and display. Require PC software and 24LC512 EEPROM.

Misc:		
SwitchBL	*20*	Switched O/P pin (primarily for backlight control).

Multi-Commands. (Points, Lines, Circles, Squares and Bargraph data).
This allows you to send extra co-ordinate information in a single command so that GLIC-K1 can plot or display a number of points, lines, circles or squares in a single command or sequence.

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

All Data to GLIC-K1 MUST BE IN SINGLE BYTES - NOT WORDS!!

PrintAT	Command Number: 100

Description:

Prints Text at Row X and Column Y. Note text screen is 20 wide by 7 high. Format:

    Serout <Pin> , T2400 , (100, X , Y , I,"My text up to 20 chars")

I = 1 means INVERSE MODE   I=0 means NORMAL MODE

Store Text	Command Number: 50

Description:

Stores text in EEPROM in a line by line format. Each line can be up to 20 chars in length.
You specify the 'line number' and can then recall it using Get and Print   AT any time. There
are a maximum number of 100 EEPROM locations for your text line.

Format:

    Serout <Pin> , T2400 , (50, Number , "My text up to 20 chars")

This will store the text in EEPROM location <Number> where number is 1 to 100.

Get Text and Print AT	Command Number: 101

Description:

Retrieves text from EEPROM location and prints at Row X and Column Y.
Note text screen is 20 wide by 7 high.
Format:

    Serout <Pin> , T2400 , (101, Number , X , Y)

This will retrieve text from EEPROM location <Number> and print at Row X , Column Y.

Graph X Axis Title	Command Number: 184

Description :

Prints text for X axis I/D after **Bar** Chart has been plotted. Max 14 Chars. Format:

    Serout <Pin> , T2400 , (184 , "My X- Title")

Graph Y Axis Title	Command Number: 185

Description:

Prints text for Y-axis caption for the Bar Chart command. Maximum 6 characters.
Format:

    Serout <pin> , T2400 , (185 , "Ytitle")

Bar Chart Main Title	Command Number: 186

Description:

Prints text for Graph Title after **Bar** Chart has been plotted. Max 18 Chars.
Format:

    Serout <Pin> , T2400 , (186,"My Graph   Title")

Command Detail

Print Page   **Command Number: 123**

Description:

Prints a page of Text recalled from EEPROM. Handy for creating a detailed Help or Information page.   Note text screen is 20 characters wide by 7 high.

Format:

>   Serout <Pin> , T2400 , (123 , Eline1, ELine2)

This will retrieve up to 7 lines of text from EEPROM between locations ELine1 and Eline2 and display it on LCD as a page.

Example:

>   Serout <Pin> , T2400 , (123 , 20 , 25)

This will retrieve text stored on EEPROM from locations 20 to 25 and print the 6 lines as a page on the GLCD.

Clear Screen   **Command Number: 10**

Description:

Clears both Text and Graphic screens. Text and Graphics are stored in separate sections of graphic memory and are, in effect, superimposed on each other.

Format:

>   Serout <Pin> , T2400 , (10)

GRAPHICAL COMMANDS

Plot (a pixel)   **Command Number: 150**

Description:

DRAWS a single pixel on the Graphic Screen. The GRAPHICS screen has a resolution of 128 pixels wide (Gx) by 64 pixels high (Gy) and is stored in a separate section to Text. Note: Zero,Zero starts at the top-left of the screen

Format:

>   Serout <Pin> , T2400 , (150 , Gx , Gy)

UNPlot (a pixel)   **Command Number: 151**

Description:

DELETES a single pixel on the Graphic Screen. The GRAPHICS screen has a resolution of 128 pixels wide (Gx) by 64 pixels high (Gy) and is stored in a separate Note: Co-ordinate Zero, Zero (0 , 0) starts at the top-left of the screen

Format:

>   Serout <Pin> , T2400 , (151 , Gx , Gy)

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

DrawLine   Command Number: 155 , 1

Description:

Draws a line on the screen between two specified co-ordinates.
This is one of a group of commands called '**multi-commands**'. This simply means that a number of co-ordinates can be included in a single command or list.

Format:

        Serout <Pin> , T2400 , (155,Set, Gx1,Gy1, Gx2,Gy2,....)

In it's simplest form this command will draw a line between Gx1,Gy1 and Gx2,Gy2.

Example:

        Serout <Pin> , T2400 , (155, 1 , 20 , 25 , 70,25)

This will draw a horizontal line between 20,25 and 70,25.
Parameter <Set> defines whether to draw or undraw (delete)
<Set>   1 draws the line.
<Set>   0 undraws (deletes) the line.

As a multi-command we can extend this in two ways.

1. Parameters within the brackets.

Example:

        Serout <Pin>, T2400, (155 , 1, 20,25,70,25,30,40,50,40,..... max 6 co-ord groups)

This will draw 2 lines :
Line 1: 20,25 to 70,25
Line 2: 30,40 to 50,40
You can draw up to 6 lines by entering the co-orinate groups within the brackets

2. Initialising the command and then loading data co-ordinates.

Example:

        Serout <Pin>, T2400 , (155 ,1)   ' Initialise Draw Line Command
        Loop:
        Calculate more X1,Y1,X2,Y2 values   ' Calc co-ordinates data
        Serout <Pin> , T2400, (X1,Y1,X2,Y2)   ' Send co-ords to GLIC
        GoTo Loop   ' Repeat Loop (6 max)

Note: If you use the second method then you must make sure that the co-ordinate data follows the initialisation well within 30mS or GLIC will assume end-of-data.

The sample PICAXE basic programme on the website shows how to use this method to draw a grid. This multi-command method can draw multiple lines far more quickly than repetitive Serout commands from PICAXE - and will save PICAXE code space.

UnDraw Line   Command Number: 155 , 0

Issue the same command structure as above but make the <set> value as zero.
Example:

        Serout <Pin> , T2400 , (155, **0** , 20 , 25 , 70,25)

This command will allow you to delete selected lines without clearing the whole screen.

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

DrawCircle(s)   Command Number: 160 , 1

Description:

Draws a circle on the screen at a specified x,y co-ordinate and radius R.

This is one of a group of commands called 'multi-commands'. This simply means that a number of co-ordinates can be included in a single command or list.

Format:

    Serout <Pin> , T2400 , (160 ,Set, Gx1,Gy1, R,....)

In it's simplest form this command will draw a line at Gx1,Gy1 with radius R.

Example:

    Serout <Pin> , T2400 , (155, 1 , 20 , 25 , 15)

This will draw a circle of radius 15 pixels at centre 20,25.

Parameter <Set> defines whether to draw or undraw (delete)

<Set>   1 draws the circle.
<Set>   0 undraws (deletes) the circle.

As a multi-command we can extend this in two ways.

1. Parameters within the brackets.

Example:

    Serout <Pin>, T2400, (160 , 1, 20,25,10, 25, 30, 40, 50, 40, 20..... max 7 groups)

This will draw 3 circles :

You can draw up to 6 circles by entering the co-orinate groups within the brackets

2. Initialising the command and then loading data co-ordinates.

Example:

    Serout <Pin>, T2400 , (160 ,1)   ' Initialise Draw Circle Command
    Loop:
    Calculate more X1,Y1, R values   ' Calc co-ordinates data
    Serout <Pin> , T2400, (X1,Y1, R)   ' Send co-ord group to GLIC
    GoTo Loop   ' Repeat Loop (6 max)

Note: If you use the second method then you must make sure that the co-ordinate data follows the initialisation well within 30mS or GLIC will assume end-of-data.

The sample PICAXE basic programme on the website shows how to use this method to draw three circles and clear them. This multi-command method can draw multiple circles far more quickly than repetitive Serout commands from PICAXE - and will save PICAXE code space.

UnDraw Circle (s)   Command Number: 160 , 0

Issue the same command structure as above but make the <set> value as zero.

Example:

    Serout <Pin> , T2400 , (160, **0** , 20 , 25 , 15)

This command will allow you to delete selected circles without clearing the whole screen.

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

DrawSquare(s)   **Command Number: 165 , 1**

Description:

Draws a square on the screen at a specified x,y co-ordinate and Edge Length EL.

This is one of a group of commands called 'multi-commands'. This simply means that a number of co-ordinates can be included in a single command or list.

Format:

> Serout <Pin> , T2400 , (165 , Set, Gx1,Gy1, EL,....)

In it's simplest form this command will draw a square at Gx1,Gy1 with radius R.

Example:

> Serout <Pin> , T2400 , (165 , 1 , 20 , 25 , 15)

This will draw a square of edge length pixels at centre 20,25.

Parameter <Set> defines whether to draw or undraw (delete)

<Set>   1 draws the square .

<Set>   0 undraws (deletes) the square .

As a multi-command we can extend this in two ways.

1. Parameters within the brackets.

Example:

> Serout <Pin>, T2400, (165 , 1, 20,25,10, 25, 30, 40, 50, 40, 20..... max 7 groups)

This will draw 3 squares :

You can draw up to 7 squares by entering the co-orinate groups within the brackets

2. Initialising the command and then loading data co-ordinates.

Example:

> Serout <Pin>, T2400 , (165 ,1)   ' Initialise Draw Square Command
> Loop:
> Calculate more X1,Y1, R values   ' Calc co-ordinates data
> Serout <Pin> , T2400, (X1,Y1, R)   ' Send co-ord group to GLIC
> GoTo Loop   ' Repeat Loop (6 max)

Note: If you use the second method then you must make sure that the co-ordinate data follows the initialisation well within 30mS or GLIC will assume end-of-data.

This multi-command method can draw multiple squares far more quickly than repetitive Serout commands from PICAXE - and will save PICAXE code space.

UnDraw Square (s)   **Command Number: 165 , 0**

Issue the same command structure as above but make the <set> value as zero.

Example:

> Serout <Pin> , T2400 , (165, **0** , 20 , 25 , 15)

This command will allow you to delete selected squares without clearing the whole screen.

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

Plot Bar Chart   Command Number: 180

Description:

Plots a bar-chart based on data supplied. Maximum of 30 byte-size values. GLIC will calculate scale, chart height and bar width automatically. This is a multi-command so that you can stream data into the GLIC-K1 from a loop.

Format:

    Serout <Pin> , T2400 , (180, CLS, Data 1,.... Data 30)

<CLS> defines whether you wish to Clear the screen first. CLS= 1 clears screen.

Example:

    Serout <Pin> , T2400 , (180, 1, 25 , 30 , 15 , 56 , 157, 200 , 120 , 85 , ...)

This will plot the bar chart based on data supplied.

Plot Bar Chart - as a Multi-Command.

Example:

    Serout <pin> , T2400 , (180 , 1)   **' Initialise Plot Bar Chart Command**
    Loop:
     Read PICAXE EEPROM   **' Read Stored data**
     Serout <pin>, T2400, (Data Byte)   **' Send data byte to GLIC-K1**
     Select new PICAXE EEPROM location
    GoTo Loop: **' Loop again - max total 30 bytes.**

Remember, as with all multi-commands used in this way, you must send data within 30mS or GLIC will think you have ended transfer. DO NOT send more than 30 data bytes!

Draw: Pointer 1 Pointer 2   Command Number: 200 / 210

Description:

DRAWS a simple line pointer on the screen at centre X,Y.
    WARNING:- ONLY TWO POINTERS SHOULD BE USED ON THE SCREEN.

Format:

    Serout <Pin> , T2400 , (200 , Gx , Gy , Length ,   Angle , Set_Clear)

<Angle>   has values of 0 to 255 (eqivalent to 0 - 360°). Value 0 is at 3 o'clock.
<Length>   has values of 1 to 50 pixel length.
<Set_Clear> 0 UNDraws a pointer, 1 Draws the pointer.

Note: If you issue the pointer command without a Clear Screen command then GLIC-K1 will automatically delete the previous pointer position to give an *animation* effect.

Example:

Serout <pin> , T2400 , (200, 60, 30, 20 , 0 ,1) ' Pointer 1 pointing at 3 o'clock

Serout <pin> , T2400 , (210, 20, 30, 20 , 60 ,1)' Pointer 2 pointing at 6 o'clock

Command Detail

In the following detail the <Pin> refers to the PICAXE pin you use to output Serial Data.

Get Bitmap and Display   **Command Number:** 221

Description:

Prints a bit-map image to the GLCD screen.
This requires prior download of a 120 x 60 pixel 1 bit bmp image from your PC.
You can store up to 4 images on the EEPROM (in addition to Text).
Don't expect photographic quality - these images will appear quite 'dotty'.
You will require GLICTalk-K PC application to perform the download and you will need to have the EEPROM connected to GLIC-K1.

Format:

    Serout <Pin> , T2400 , (221 , <Image No.>)

<Image No> defines the image number to be displayed.

Example:

    Serout <Pin> , T2400 , (221 , 2)

This will print Image Number 2 to the screen.

Plot Border   **Command Number: 152**

Description:

Prints a single width line border around edge of Graphic screen

Format:

    Serout <pin> , T2400 , (152 , <CLS>)

  defines whether screen is cleared first. 1 = Clear   0 = Leave alone.

Example:

    Serout <pin> , T2400 , (152 , 0)   ' Initialise Plot Bar Chart

This will generate a border around the edge of the screen.

Switch Backlight   **Command Number: 20**

Description:

Switches GLIC Pin Number 3 High / Low. CANNOT be used to power backlights directly. You **must** use a transistor to handle the backlight (or any other load) current.
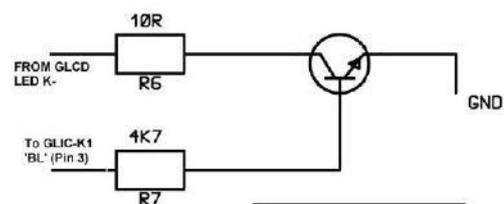
Format:

    Serout <pin> , T2400 , (20 , <CTL>)

<CTL>   1 = On/High   0 = Off/Low.

Example:

    Serout <pin> , T2400 , (20 , 0)   ' Switch pin OFF

Switches pin OFF/Low. Remember, this is a sink/source pin. Max 20mA or else POP!!

Special Characters.

In addition to the standard   ASCII/ANSI character set there are some additions pre-programmed. Don't expect them to be PC quality as they are only based on a 6x8 pixel block.

These are block graphic characters as used by early Pcs.

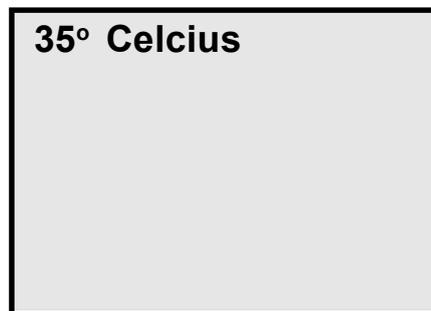ASCII Range for text-block graphics is 1 to 25

These can be only used with the Print group of instructions.

ASCII character number 26 is the degree symbol - as in 35°C

Example:

    Serout <pin number> , T2400 , (100 , 0 , 0 , "35" , 26 , " Celcius")

    This will print:-

```
35° Celcius
```

Try and improve the 'readability' of your code. Look at this example for the above.

In your PICAXE code, you could write:

    Symbol Degr   = 26
    Symbol PrintAT   = 100

Then, later in your code:

Serout <pin number> , T2400 , (PrintAT , 0 , 0 , "35" , Degr ," Celcius")

This produces the same result as in the example command above - but is clearer.

The use of Symbol aliases for all your commands will make your code much easier to read when you come back to make changes at a later date.

Remember, aliases such as 'Degr' (or numbers such as 26), they are just numbers. When used in print statements it is assumed that the number used corresponds to an   ASCII character. If used in any other statements they are just data/numbers.

Things to Remember:

1. All commands numbers & data must be single BYTE values - NOT words.

2. Commands can be variables or constants set using the 'Symbol' command.

See '*Suggestions on Code Writing*' for more ideas.

3. When using multi-commands do not have any long delays between data transmission.

If you use the multi-command format where you Issue a command and *then* follow it with data, you MUST make sure that each data byte is sent within 30mS of the previous data byte. Otherwise GLIC-K1 will assume you have finished and execute the instruction. Data sent to GLIC-K1 whilst it is still executing an instruction will usually be ignored, BUT if it went in at the wrong moment it could temporarily upset the Instruction sequence - with strange results!

4. GLIC takes a finite amount of time to process your command.

Each time you send a command and data to the GLIC i.c. it takes a certain amount of time to process the data and perform the action. If you have the 'STATUS' pin connected to a PICAXE input pin, you can check the level before issuing a new command with an 'IF' statement.   **High** = Ready to Receive your Command.   **Low** = Busy.

If you don't use the 'STATUS' pin you will have to allow a period for the GLIC to work. Some commands take longer than others. Anything which involves writing to the EEPROM or multi-commands will take a little longer. Allow between 150mS and 350mS to process before issuing a new command.

However, this little delay needn't be simply a BASIC 'Pause' statement. If your PICAXE has other things to do then you can do them during this period - this can be part of the overall delay. Try it out, you won't break it.

5. Do NOT send more data than the command requires.

This may briefly upset GLIC's instruction sequence and this may produce odd results.

6. Power Up recommendations.

The GLCD takes up to a second to start up and GLIC allows for this before clearing the registers. However, it is recommended that if your PICAXE controls the power then you leave a real PAUSE of 1500mS before issuing a command.

It is also recommended that your **first** command is to clear the text/graphic screen:

Serout <pin> , T2400 , (10) : Pause 300   ' Clear GLCD and a small pause.

This also applies if you include a reset switch / or the reset circuit.

Reserved Command Numbers: 33, 34, 147 , 149, 220   DO NOT USE THESE!

Suggestions for Code Writing:

' 1. Set GLIC Command Numbers as Constants.

```
symbol ClearLCD  = 10                          ' Clear LCD Screen
symbol StoreText = 50                          ' Store Text (max 30 chars) in EEPROM
symbol PrintAT   = 100                         ' Print string at Xt,Yt
symbol GetText   = 101                         ' Get from EEPROM and Print AT Xt,Yt
symbol PrintAW   = 120                         ' Print AT with Autowrap (one line only)
symbol DoCircle  = 160                         ' Draw a circle
symbol DoSquare  = 165                         ' Draw a square
symbol DrawLine  = 155                         ' Draw Line or Lines
symbol PlotPix   = 150                         ' Plot pixel at Xg,Yg
        etc....
```

2. To make sure GLIC / GLCD initialises properly, always start with a Clear Screen.

```
' GLIC Startup pause and clear on PICAXE 18X
Pause 1500                                     ' Allow GLCD to settle after power-up
Serout 7,T2400, (ClearLCD) : Pause 200         ' Issue Clear Screen and a pause
```

3. If you don't use the STATUS connection you must include pauses after Commands.

If, of course, your programme carries out other tasks between GLIC Serout commands then you adjust/remove the pauses as necessary. It is the OVERALL delay between GLIC commands that it is important.

```
'[MAIN CODE BEGINS] on PICAXE 18X   ***************************************************

  serout 7,T2400, (ClearLCD)
   pause 200                                   ' Have a little pause
   serout 7,T2400, (Border,0)                  ' Do Border.  DON'T clear GLCD
   pause 500                                   ' A pause

   serout 7,T2400, (PrintAT,1,5,"Hello PICAXE")   ' PRINTAT (Format 100,X,Y,n,n)
   pause 250                                    ' A pause


' Multi-commands require slightly longer overall delays.
   serout 7 , T2400, (DoCircle,0,60,30,20)      ' Use a ZERO to clear the circle
   pause 500
```

The pause values shown in the DEMO PICAXE Basic programme are much larger than needed and are there so that the operation can be clearly seen.

Pauses ARE required if the 'STATUS' line from GLIC-K1 is not used with appropriate check code. If using the STATUS connection then a simple   IF <Input Pin> = 1 THEN ...   structure can be used.

GLIC-K1 software is being continuously improved and is subject to change without notice.

FGC

Addendum. A1.

NEW COMMAND:

| FILLED SQUARE | Command Number: 170 |

Description:
Produces a square filled or unfilled.

Format:
> Serout <pin> , T2400 , (170 , Set , Xc , Yc , Edge)

> **Set**   1 = Filled   0= Unfilled
> **Xc**   Centre X co-ordinate
> **Yc**   Centre Y co-ordinate
> **Edge** Edge length in pixels

NOTE: Graphics area is 128 pixels wide by 64 pixels high.

Example:
> Serout 7 , T2400 , (170 , 1 , 80 , 30 , 20)

This will create a solid box of size 20 x 20 pixels at centre X=80 and Y=30.

More readable coding example:-
Near top of code:
> Symbol FilledBox = 170
> .
> .

Then, when required:
> Serout 7, T2400 , (FilledBox , 1 , 80 , 30 , 20)
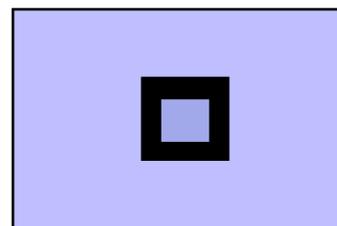
0 , 0

128 , 64
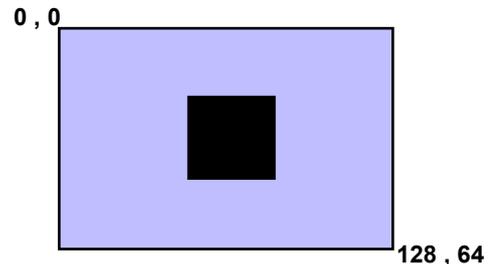
By using the 'unfill' setting we can hollow out a square:
Example:

After creating the solid square:-

> Serout 7, T2400 , (FilledBox , 0 , 80 , 30 , 15)

Zero = Unfill
(Background colour)

NEW FUNCTION:

Display Version Number.
Hold 'F1' to Vdd (+5V) and 'reset' GLIC (MCLR to Ground briefly).

A border animation will run and then display the software version.

FGC GLIC-K
Graphic
Interpreter
Vx.y MMYY

FGC

Addendum. A2.

Programming Notes.

Subject: Use of STATUS pin output.

The status pin level can be used to make sure that you don't send overlapping commands. i.e. sending a command before the preceding command has been executed by GLIC-K.

If commands overlap then the GLIC-K behaviour will be unpredictable - undesirable.

There are three basic methods of preventing command overlap.
1. Simple 'pause' instructions to allow GLIC-K command to finish before proceeding.
2. Testing the 'Status' pin.
3. Executing subsidiary code and then testing the 'Status' pin.

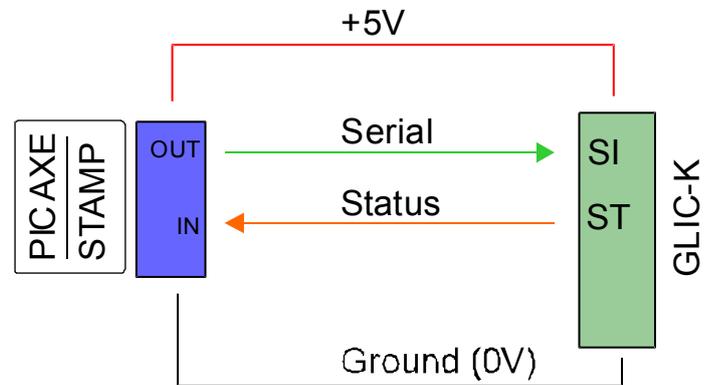In this example you will need to connect your PICAXE / Basic in the following way.

Connection Summary.

Common Ground

Common +5V supply

PICAXE/Stamp OUT -- GLIC 'SI' (Serial IN)
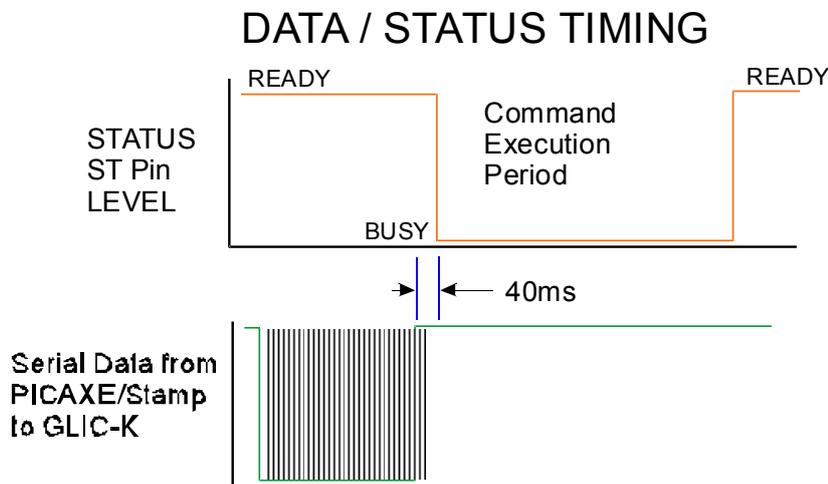
PICAXE / Stamp IN   --- GLIC 'ST' (STATUS)

The function of STATUS is this:-

HIGH means GLIC is Ready to Receive Data
LOW means GLIC is BUSY, so don't send more command data until ready.
When GLIC receives a valid command/data the STATUS will go low while it is being processed.

DATA / STATUS TIMING

STATUS ST Pin LEVEL

READY — Command Execution Period — READY — BUSY

40ms

Serial Data from PICAXE/Stamp to GLIC-K

This diagram shows the STATUS pin level after valid data has been sent to GLIC-K1.

As you can see the STATUS pin goes low (Busy) whilst it is processing the information and carrying out the command.

Once the command has been executed it returns to the High (Ready) state.

You can use this feature in your PICAXE / Stamp code to prevent command overlap.

Making use of the Status pin in PICAXE / Stamp code.

See the previous page to see how the Status pin level changes when 'Ready' and 'Busy'.

We can use this to prevent GLIC Command overlap.

If you DO NOT want to use the Status level to check (perhaps you haven't a spare PICAXE / Stamp pin) you will have to use pauses to prevent Command Overlap.

Code Example (PICAXE) to use the STATUS pin.
(This assumes you have connected the STATUS pin as shown on previous page.)

```
Serout SO,T2400,(PrintAT,0,2,0," Use Status")
do
loop until GStatus=1
```

The above code will sit in the DO-LOOP until Status is set HIGH by GLIC.

If you have code which is doing a number of operations you can use a similar arrangement but execute code while waiting.

```
serout SO,T2400,(PrintAT,0,4,0," Parallel ")
' Mycode here...
' Could be reading an ADC
' or some calculations while GLIC
' is dealing with the command.
do
loop until GStatus=1
```

This can allow a type of parallel processing for faster coding. Obviously the extra code (in **green** above) must **NOT** carry out any GLIC commands.

These examples were written for the Version 5.0.x PICAXE Programme Editor.

For a download of the Demonstration code (PICAXE) click this link (online):-
www.fgcvme.co.uk/Demo_Status_Pin.bas

For older versions you will have to use IF .. THEN .. GOTO syntax.

Coding for Basic Stamp will be similar.

GLCD Backlight LED Limiting Resistor.

On the GLIC-K1 I/F pcb the component **R14** limits the LED current.

It is crucial that you should read the GLCD Data Sheet section which specifies the maximum LED current.

If you are using the Fordata Blue/White GLCD module as supplied by Revolution Education Ltd the maximum backlight LED current is 40mA.

We would recommend a slightly lower current and would strongly suggest a value of 33R (33 ohms) to be used. This will give a backlight LED current of about 25 to 27mA.

For other makes you must consult the appropriate Data Sheet noting that older yellow/green GLCDs tend to use more current. Note that the BC182L transistor used has a maximum continuous rating of 100mA. If you have used a BC184L this can handle slightly higher currents.

Inverter -V supply for GLCD.

The negative voltage supply is necessary to make the GLCD pixels visible.
Most KS0108 based GLCDs (including the Rev-Ed supplied GLCD) have an on-board negative V generator. This means that components C2 , C3 , C4 and IC1 can be omitted. You must ensure that GCLD pin 18 (-VE voltage output "V-LCD") is connected to the 'contrast' potentiometer.

If you are using the FGC GLIC-K1 I/F pcb this simply means fitting the jumper across the contacts marked 'GLCD'.

GLIC-K1 connection to use GLCD's -V generator.