DiosPro 28/40 Interface
to SparkFun and Crystalfontz 128x64 Graphic LCD

Character based LCD's are great for presenting the user with textual information. However there are times you need a bit more freedom when presenting data. In this application note I will present a interface between the Dios chips and a Crystalfontz 128x64 graphic LCD.

Compatible Displays

This application note is compatible with the following displays:

Directly compatible
•SparkFun 128X64 STN LED # LCD-00710 (Best Buy)
•Crystalfontz CFAG12864B-YYH-V
•Crystalfontz CFAG12864B-TMI-V
*Jameco #658953 (Needs Slight Board Mod)

Backlight provided by external power source.
•Crystalfontz CFAG12864B-WGH-V

Display pinout will differ from Above
•POWERTIP PG12864ERS-INN-H

Other
•Displays controlled by KS0107/KS0108
•Displays controlled by HD61202

Note that the Crystalfontz and Graphical LCD's listed have negative voltage generators built-in to provide the contrast voltage needed to drive the LCD. Many other graphical LCD's do not and will require a special inverter to provide this power. These are not covered in this application note.

KS0107/KS0108 Overview

The displays listed above are controlled by the KS0107/KS0108 or compatible controller. This is a very popular controller and has made its way into many Graphic LCD's.

In many cases the actual Graphic LCD will have multiple controllers stacked to create larger display sizes. This paper is primarily concerned with those displays with dual controllers.

When writing to the controllers the CS1 and CS2 pins will be selected to turn on the appropriate controller. It's also possible to access both portions of the display at once. This is particularly nice when initializing and clearing the display.

The KS0107/KS0108 does not have a character generator so this must be provided in software.

Dios Library Overview

As popular as the KS0107/KS108 controller is, It does have its negatives. For one just about every piece of documentation I have seen regarding this controller has terrible errors and are very hard to follow. Also the nomenclature used seems to be counter intuitive.

For this reason I have created a set of libraries to access the level memory of the LCD's. Iwill be using my on nomenclature.

It's not necessary that you understand the low level technical lay - out of the libraries or the controller. I am simply presenting this information so you can expand upon the library if you wish.

Display layout (low level)

Display pages the display is divided up into two sections I call pages.

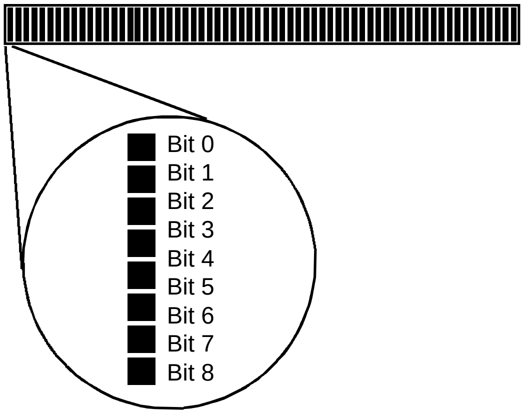
Page 1	**Page 2**

Each page can display 64x64 pixels and is controlled by its own controller. Many of the lower level library commands will contain a parameter for selecting the appropriate page.

Each page has 8 lines numbered 0-7.

Page 1	Page 2
Line 0	**Line 0**
Line 1	**Line 1**
Line 2	**Line 2**
Line 3	**Line 3**
Line 4	**Line 4**
Line 5	**Line 5**
Line 6	**Line 6**
Line 7	**Line 7**

Each line has 64 positions that contain a 1 pixel wide by 8 pixel tall strip.



These strips can be written to or read from with the low level commands.

Display layout (high level)

At the high level we don't concern our selves with thepages and strips. We simply issue a command and the inner workings are handled for us.

Character Display

First lets look at the software character generator layout. The characters are 6x8 with the only the 5x7 portion of the character showing.

The display has 8 lines of 20 character data. An internal pointer keeps track of the current position and is incromented with each character sent to the display. If the pointer reaches the end of the line the character will be placed on the next line.

Character line 0 (20 character positions)
Character line 1 (20 character positions)
Character line 2 (20 character positions)
Character line 3 (20 character positions)
Character line 4 (20 character positions)
Character line 5 (20 character positions)
Character line 6 (20 character positions)
Character line 7 (20 character positions)

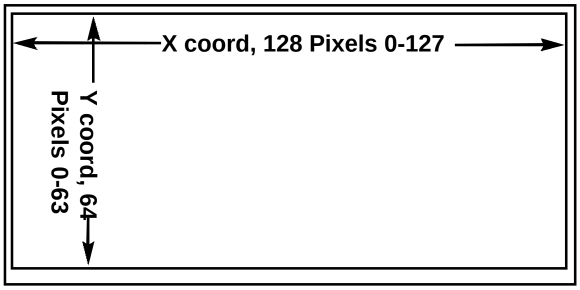
Note that there are 4 pixels to the right and 4 pixels to the left of the character display area. This area is not touched when characters are displayed. This area could be used for small menu pointers or small bullets indicating a selected line of text.

There are a total of 160 character positions that can be written to. You will use the **GLCDwritechar** and **GLCDwritecharxy** commands to write to these positions.

Pixel Display

The software pixel generator commands will automaticaly take care of selecting the correct page,line,pos and bit in the strip.

The routines read in the current strip where the pixel is located and set or reset the bit as indicated then write the modified strip back out to the display.



The pixels are arranged with the x coordinate along the width with the lowest pixel at the left (0) and the highest at the right (127).

The y coordinate is aligned with the height with the lowest pixel at the top (0) and the highest at the bottom (63).

You write to the pixels with the **GLCDsetpixel** command. This command is independent of the character display system and can display right over the top of the characters.

Hookup

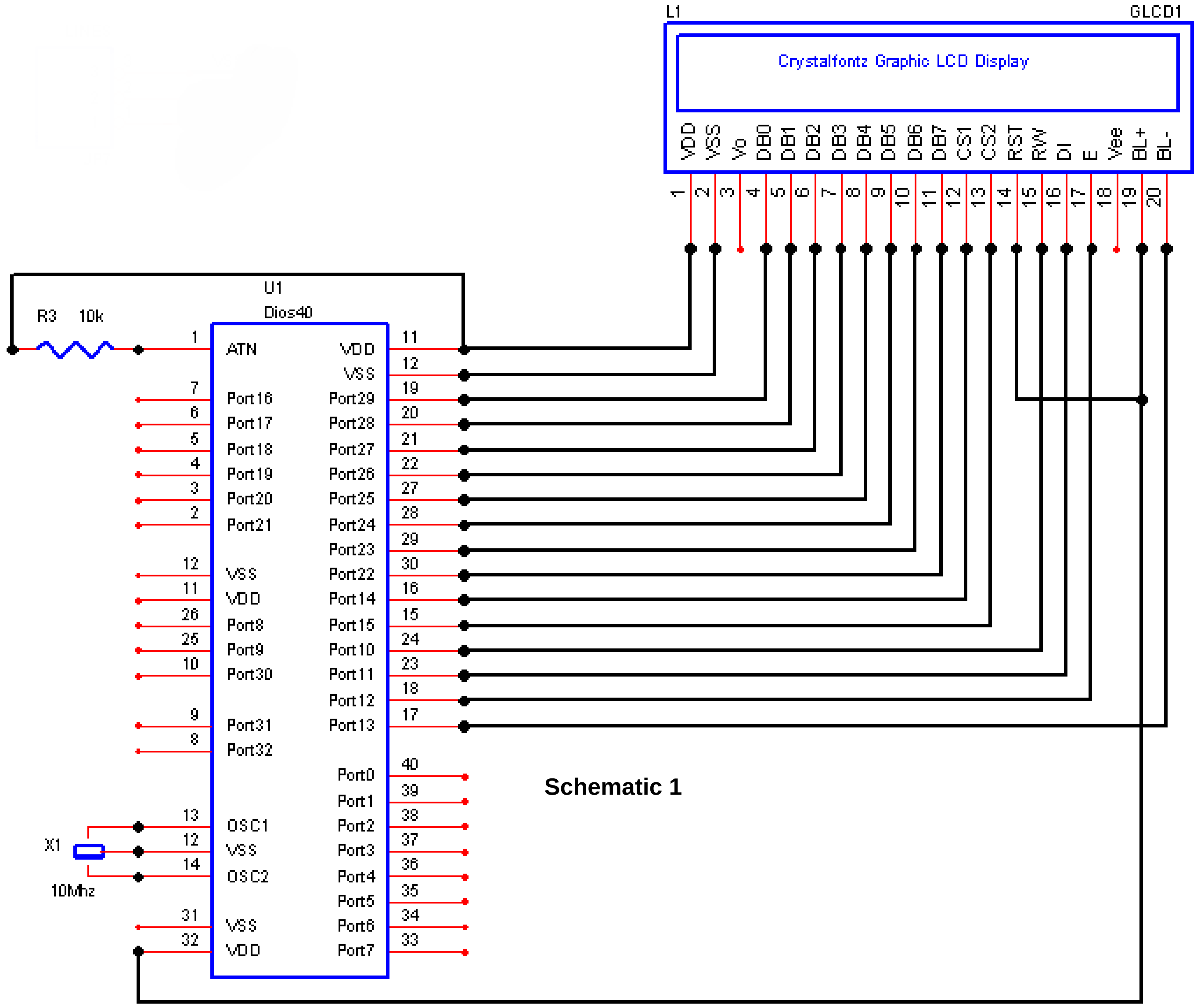
The Dios Universal LCD carrier was designed as the perfect interface to the Crystalfontz CFAG12864B-YYH-V and Crystalfontz CFAG12864B-TMI-V. These two graphic LCD's will plug directly into the Universal connector located on the board.

You can also connect various other graphic LCD's using the jumpers on the Universal connector. Please refer to the Dios Universal LCD carrier hookup manual for more information.

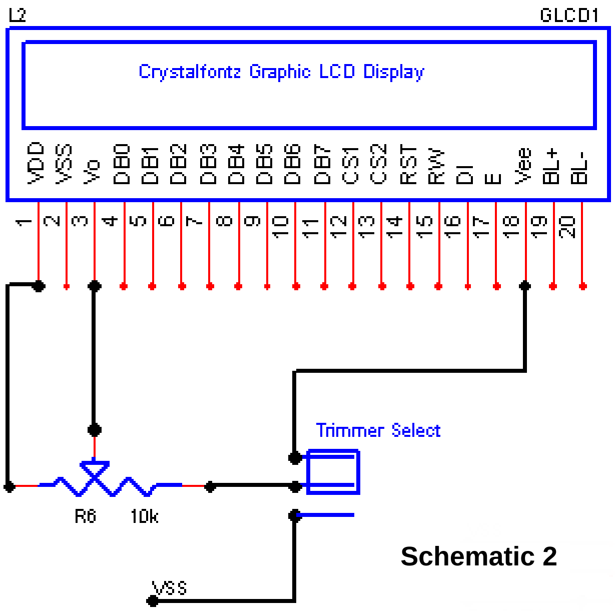
The following are the schematics of the various componets on the PCB.

- Schematic 1Graphic LCDHookup
- Schematic 2Graphic LCD Contrast Hookup
- Schematic 3Power
- Schematic 4RS232 interface
- Schematic 5Header and Jumpers

Please refer to the Dios Universal LCD Carrier for more information and details on the various jumpers and headers.



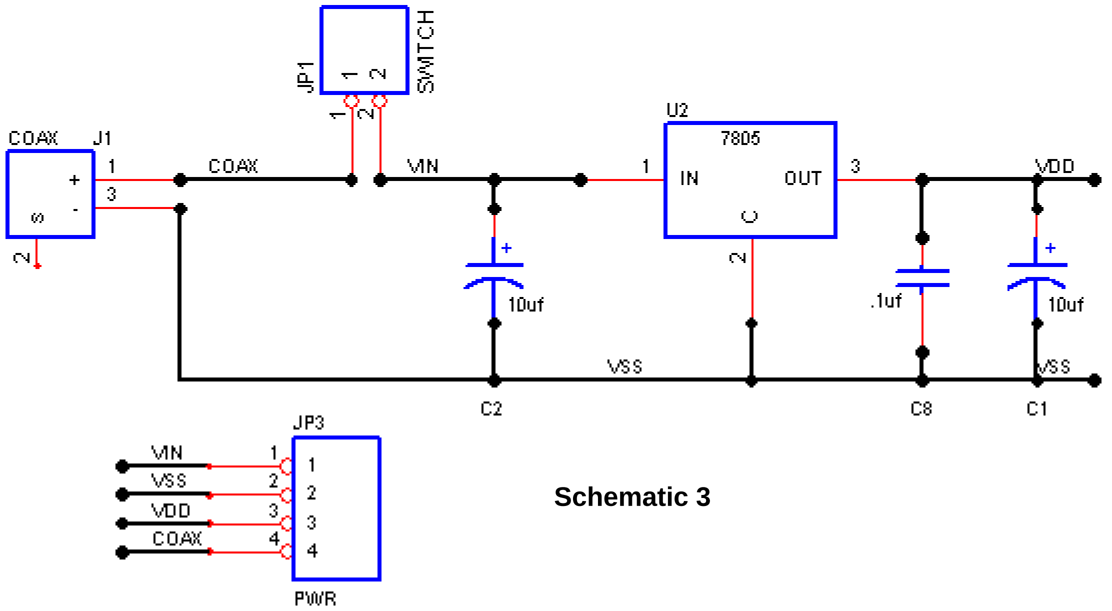
Schematic 1



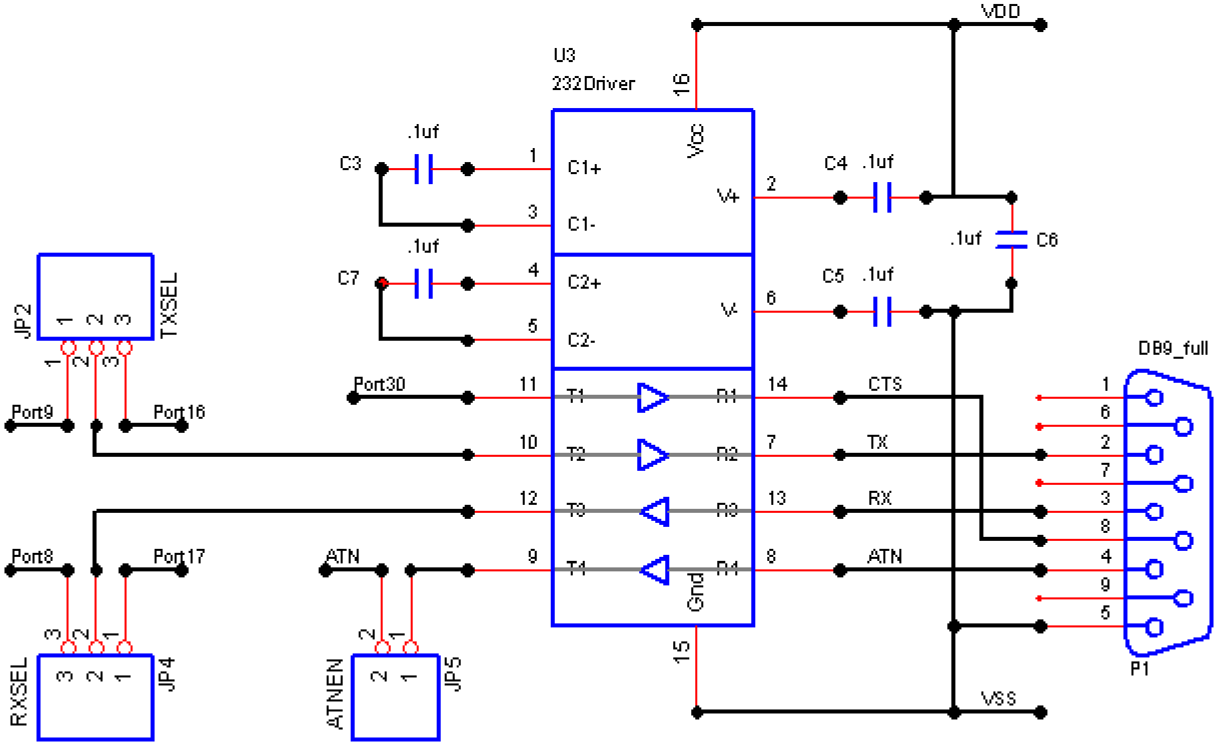
Schematic 2

Notice the Trimmer Select jumper is always in place as shown when using graphic display's.

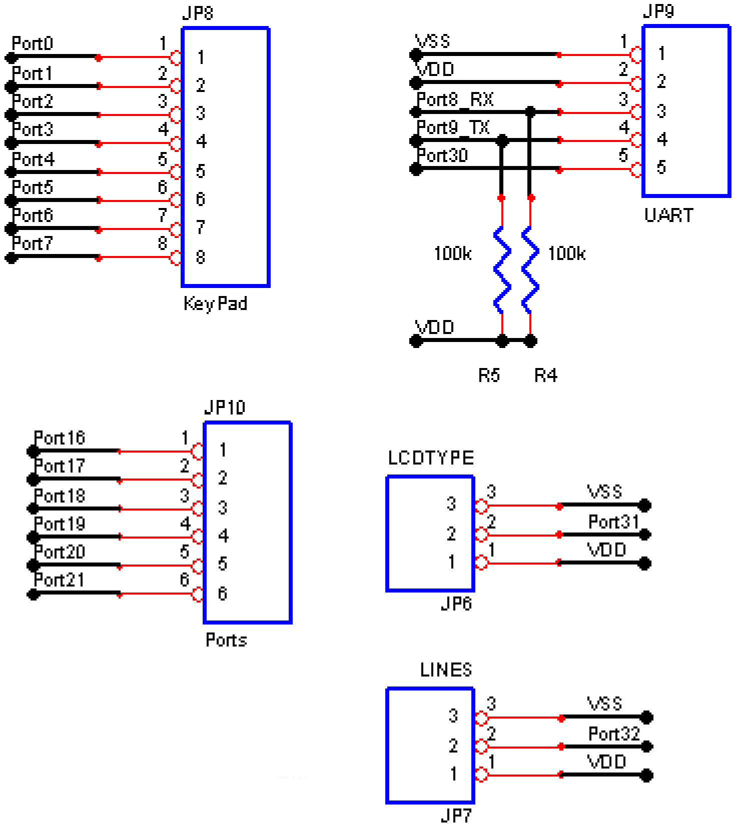
This jumper routs the graphic display's onboard negative voltage generator to the trimmer potentiometer.



Schematic 3



Schematic 4



Schematic 5

Software Library

The library is included with Dios Compiler versions 2.0.10 and greater. The use of any of the following commands will automaticly add the Library to your source file.

High Level Character Commands

These high level commands are used to write characters or initilize the display.

GLCDinit / GLCDinit2 **Initilize Graphic LCD**

syntax GLCDinit / GLCDinit2

Description

The GLCDinit command is used to setup the ports and initialize the graphic LCD. It must be the first graphic LCD command called. See **Example 1**

Use GLCDinit2 for SparkFun and Jameco graphic LCD's

Dios - Graphic LCD

GLCDLED — Set Brightness level of the backlight LED

syntax: GLCDLED Level

Operands: **Level**: Integer value 0-255 that sets the LED brightness level.

Description: This command will allow you to control the backlight LED. In order for this command to work the LED negative lead must be connected to port 13 on the Dios. The positive lead should be connected to Vdd. See **Example 1**

GLCDchar — Write text to the current position on LCD

syntax: GLCDchar Table,Text

Operands: **Table**: Table pointer, This sets the character table to use when writing characters to display. currently the only valid table is called chars.

Text: Quoted text or string. These are the characters to print.

Description: This command writes one or more characters to the LCD at a particular position. The characters will wrap automaticaly.

GLCDcharpos — Write text to the LCD display at pos X and Y

syntax: GLCDcharpos Table,Line,Pos,Text

Operands: **Table**: Table pointer, This sets the character table to use when writing characters to display. currently the only valid table is called chars.

Line: Integer value 0-7, This is the line to start printing on.

Pos: Integer value 0-20, This is the position on the line to start printing on.

Text: Quoted text or string. These are the characters to print.

Description: This command writes one or more characters to the LCD at a particular position. The characters will wrap automaticaly. See **Example 1**

GLCDclear — Clears the whole LCD display

syntax: GLCDclear

Description: This command will clear the LCD.

GLCDfill — Fills the whole LCD display

syntax: GLCDfill

Description: This command will write all pixels black.

GLCDsetcharxy	**Sets the character position for the next character**
syntax	GLCDsetcharxy Line,Pos
Operands:	**Line**: Integer value 0-7, This is the line to start printing on. **Pos**: Integer value 0-20, This is the position on the line to start printing on.
Description:	This commands lets you set position for the next character to display.

GLCDcharraw	**Sends a character to the display at the current position**
syntax	GLCDcharrawl Table,char
Operands:	**Table**: Table pointer, This sets the character table to use when writing characters to display. currently the only valid table is called chars. **Char**: Integer value 0-255, This is the character you wish to display
Description:	This commands lets you send a single character to the display. It will display at the current position.

High Level Pixel Commands

These high level commands are used to write pixel data to the LCD display.

GLCDsetpixel	**Sets a pixel on the display**
syntax	GLCDsetpixel x,y,mode
Operands:	**x**: Integer value 0-127, This is the x position on the display of the pixel you wish to change. **y**: Integer value 0-63, This is the y position on the display of the pixel you wish to change. **mode**: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.
Description:	This commands lets you set/reset/Xor a pixel on the display. See **Example 1**

GLCDbox	**Draws a box on the display**
syntax	GLCDbox x1,y1,x2,y2,mode
Operands:	**x1**: Integer value 0-127, This is the starting x position of the box you wish to display. **y1**: Integer value 0-63, This is the starting y position of the box you wish to display. **x2**: Integer value 0-127, This is the ending x position of the box you wish to display. **y2**: Integer value 0-63, This is the ending y position of the box you wish to display. **mode**: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.
Description:	This commands lets you draw a box on the display.

GLCDboxfill

Draws a filled in box on the display

syntax

GLCDboxfill x1,y1,x2,y2,mode

Operands:

x1: Integer value 0-127, This is the starting x position of the box you wish to display.

y1: Integer value 0-63, This is the starting y position of the box you wish to display.

x2: Integer value 0-127, This is the ending x position of the box you wish to display.

y2: Integer value 0-63, This is the ending y position of the box you wish to display.

mode: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.

Description:

This commands lets you draw a box on the display. It is the same as the GLCDbox command except it is filled in.

GLCDline

Draws a line between two points on the display

syntax

GLCDline x1,y1,x2,y2,mode

Operands:

x1: Integer value 0-127, This is the starting x position of the line you wish to display.

y1: Integer value 0-63, This is the starting y position of the line you wish to display.

x2: Integer value 0-127, This is the ending x position of the line you wish to display.

y2: Integer value 0-63, This is the ending y position of the line you wish to display.

mode: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.

Description:

This commands lets you draw a line between two points on the display.

GLCDhline

Draws a horizontal line between two points on the display

syntax

GLCDhline x1,y1,y2,mode

Operands:

x1: Integer value 0-127, This is the starting x position of the line you wish to display.

y1: Integer value 0-63, This is the starting y position of the line you wish to display.

y2: Integer value 0-63, This is the ending y position of the line you wish to display.

mode: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.

Description:

This commands lets you draw a horizontal line between two points on the display. It can only draw horizontal strait lines but is much faster then the GLCDline command.

GLCDvline

Draws a vertical line between two points on the display

syntax

GLCDvline x1,y1,y2,mode

Operands:

x1: Integer value 0-127, This is the starting x position of the line you wish to display.

y1: Integer value 0-63, This is the starting y position of the line you wish to display.

x2: Integer value 0-63, This is the ending y position of the line you wish to display.

mode: Integer value 0-2, The mode sets the way the pixel will be drawn. 0 = Clear Pixel, 1 = Set Pixel, 2 = Xor Pixel.

Description:

This commands lets you draw a vertical line between two points on the display. It can only draw vertical strait lines but is much faster then the GLCDline command.

High Level Strip Commands

These high level commands are used to display a 1 bit wide by 8 bit tall strip on the display.

GLCDstrippos

Draws a strip on the display

syntax

GLCDstrippos Line,pos,Strip

Operands:

Line: Integer value 0-7, This is the line to display the strip on.

Pos: Integer value 0-127, This is the position on the line to place the strip on.

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description:

This commands lets you draw a single 1x8 bitmap on the display.

GLCDpagestrip

Draws a strip on the display at current position

syntax

GLCDpagestrip Page,Strip

Operands:

Page: Integer value 0-3, 0=dont display, 1=page 1, 2 = page 2, 3 = both pages

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description:

This commands lets you draw a single 1x8 bitmap on the display at the current position. Not that it is page dependant.

GLCDpagestrippos

Draws a strip on the display at a particular position

syntax

GLCDpagestrippos Page,Line,pos,Strip

Operands:

Page: Integer value 0-3, 0=dont display, 1=page 1, 2 = page 2, 3 = both pages

Line: Integer value 0-7, This is the line to display the strip on.

Pos: Integer value 0-63, This is the position on the line to place the strip on.

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description:

This commands lets you draw a single 1x8 bitmap on the display Note that it is page depend

GLCDsetstrippos — Sets the display position of the next page strip

syntax — GLCDsetstrippos Page,Line,pos,Strip

Operands:

Page: Integer value 0-3, 0=dont display, 1=page 1, 2 = page 2, 3 = both pages

Line: Integer value 0-7, This is the line to display the strip on.

Pos: Integer value 0-63, This is the position on the line to place the next strip on.

Description: This command sets the display position for the next page strip display. This command is page dependent.

High Level Fill Commands

These high level commands are used to fill in locations on the display.

GLCDfillpageline — Fills a line on the display with a particular strip

syntax — GLCDfillpageline Page,Line,Strip

Operands:

Page: Integer value 0-3, 0=dont display, 1=page 1, 2 = page 2, 3 = both pages

Line: Integer value 0-7, This is the line to display the strip on.

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description: This command fills a line on a page with a particular strip.

GLCDfillpage — Fills a page with a strip

syntax — GLCDfillpage Page,Strip

Operands:

Page: Integer value 0-3, 0=dont display, 1=page 1, 2 = page 2, 3 = both pages

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description: This command fills a complete page with a strip.

GLCDfilldisplay — Fills the display with a strip

syntax — GLCDfilldisplay Strip

Operands:

Strip: Integer value 0-255. This is the strip data. 1 bit wide by 8 bits high bit map.

Description: This command fills the complete display with a strip.

GLCDclear **Clear the entire display**

syntax GLCDclear

Description: This command clears both pages at once.

GLCDfill **Fill the entire display**

syntax GLCDfill

Description: This command fills both pages with solid

Example 1: GLCDLED, GLCDinit, GLCDsetpixel and GLCDcharpos code example

```
'Crystalfontz KS0107/KS0108 Grahic LCD
func main()
   dim x,y

   GLCDinit   '(Use GLCDinit2 for Sparkfun or Jameco)
   GLCDLED 200 'Turn on LCD backlight

   for x = 0 to 127
     GLCDsetpixel x,0,1
     GLCDsetpixel x,63,1
   next

   for y = 0 to 63
     GLCDsetpixel 0,y,1
     GLCDsetpixel 127,y,1
   next

   GLCDcharpos chars,3,4,"Hello World"

endfunc

include \lib\DiosUGLCD.lib
```

Final Thoughts

With the Dios built-in string and floating point variables the functions that could be added to this library are astounding.

I have even been playing with the idea of adding a graphic display to the KRAnalyzer to create a small compact self contained data scope.

If you come up with any functions you feel would contribute to the graphic LCD library gives us a shout.

The Dios Universal LCD Carrier can be found at ..... http://www.kronosrobotics.com/downloads/GraphicLCD.pdf

Related Products

Dios Universal LCD Carrier	http://kronosrobotics.com/xcart/customer/product.php?productid=16410
DiosPro 40	http://kronosrobotics.com/xcart/customer/product.php?productid=16428
9 Pin Cable	http://kronosrobotics.com/xcart/customer/product.php?productid=16259
SparkFun Graphic LCD	http://www.sparkfun.com/commerce/product_info.php?products_id=710
Crystalfontz Graphic LCD	http://www.crystalfontz.com/products/index-grph.html