
Digole Serial:UART/I2C/SPI Character/Graphic LCD/OLED Display Module User Manual

(last updated: Mar. 3rd 2014)

This manual will describe most common features for our Serial LCD/OLED displays and modules, each particular product may have different looks, size and material, but all interface to your master circuits and control commands are same, that means you can switch Digole Serial display in your application without any modification of your master circuit and software.

Our Serial display products are listed in figure-1, you can purchase them with lowest price at: <http://www.digole.com/index.php?categoryID=153>

What benefits you if using these products in your electronic projects?

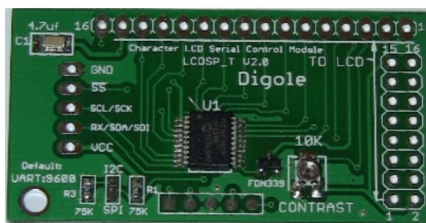
- **Save lots of the I/O resources:** these products only need 1 to 3 I/O pins from your master controller that depends on the communication type you want.
- **Easy to use:** the commands sending to products are easy to remember and understand.

On Graphic serial products:

- **Save huge memory space** to store font and start screen on graphic display: in graphic product, there are 7 preloaded fonts ready for your application, and also have 16KB memory space for your user fonts, once you uploaded the start screen or user fonts, it will be stored in products.
- Using user fonts function, you can display any graphs or characters in any language
- These products already **integrated graphic functions** such as: draw line/rectangle/circle/image, send few bytes of instruction to products, it will do it for you, that also saves your lots of code space
- You can display contents in 4 different directions: 0°, 90°, 180°, 270°(clockwise) on same screen, the product will map the coordinate accordingly.

FEATURES:

- | | |
|---|---|
| <ul style="list-style-type: none">• Communication mode: UART/I2C/SPI, detect your setting automatically• Receiving buffer: 64/256 bytes• Work with all microcontroller and microprocessor• Communication signal can work on 3.3V and 5.0V TTL• Default setting: UART baud 9600bps, I2C 0x27 address | <ul style="list-style-type: none">• Low power consumption: less than 4mA (for adapter only, completed module may be higher depends on the backlight power consumption)• Simple command sets, easy to remember• Simple graphic engine integrated (Graphic Products)• 7 preloaded fonts, font's data structure full compatible with U8Glib(Graphic Products)• UART baud (bps): 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200 |
|---|---|



Character Serial LCD Adapter V2

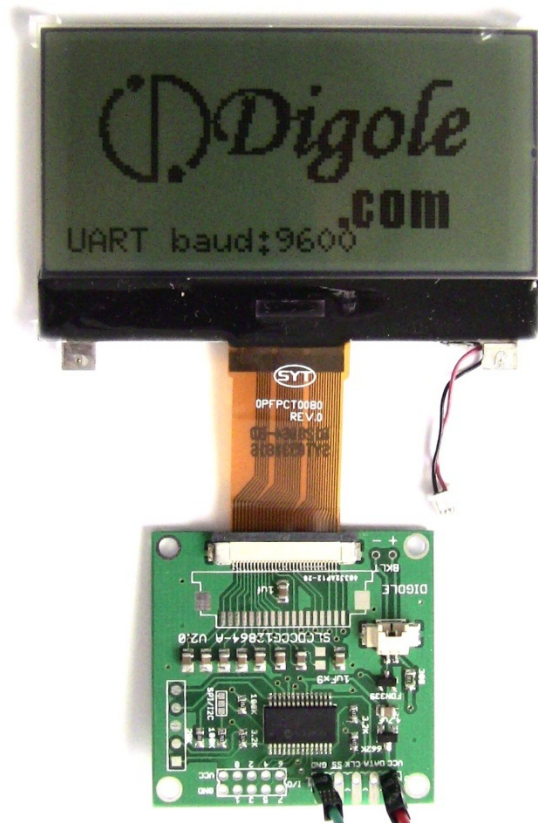


240x64 Dots Serial LCD



128x64 Serial OLED(Blue/White, 1"/1.3")

2 of 9



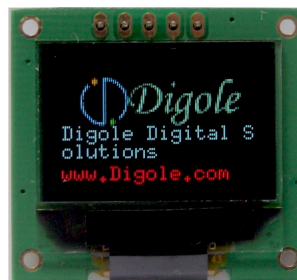
[Medium Size \(7x4.6cm\) 12864 Serial LCD+Backlight](#)



[Small size\(4.6x3cm\) 12864 Serial LCD+Backlight](#)



[1.8" 160x128 True Color Serial OLED Module](#)



[0.96" 96x64 65K Color Serial OLED Module](#)

What are adapters used for?

Character adapters can work with most 1602,1604,2002,2004 and 4002 character LCDs.

The Universal Graphic adapter work with 128 x 64 dots LCD, which LCD controller is ST7920 or KS0108 or ST7565.

We didn't sell adapters with a LCD due to you might already have LCDs or can easy to get one at low price from somewhere, so this way gives you more flexible options on your project.

How to set up the communication mode?

There are 3 different communication modes on all products: UART, I2C and SPI, what you need is just use solder to short the I2C/SPI jumper on adapter and make it works at I2C or SPI, if both jumpers are open, it works at UART,



you can find a similar jumper like this: on board.

Protocols:

- UART : 8-N-1, 8bits, No parity bit, 1 stop bit.
- I2C: Slave Mode, 7-bit address, default address is Hex:27, change able. This mode may give you a headache due to more signal options in I2C, but we make it works as standard, you just need setup your I2C on master controller as Standard Master Mode.
- SPI: 8-bits, MSB first, data on raise edge of SCK sampled; this is Standard setting on SPI too.

Character/Graphic Display Shared Command: (B-one byte, B...-Bytes)

Command	Description	Arduino lib function	note	Char Display	Mono-Graph	Color-Graph
CL	CL ear screen and set the display position to first Column and first Row (x=0.y=0), for graphic LCD, it also set the font to default and turn off the cursor.	clearScreen();	The module will not execute this command until other command received.	✓	✓	✓
CSB	set CurSor on/off	enableCursor(); disableCursor();	B=0 off, B=1 on	✓	✓	✓
BLB	Set Back Light ON/OFF, B=0 or 1, 0 off, 1 on	backLightOn(); backLightOff();	unavailable on Character Adapter V1.x	✓	✓	✓
SOOB	Set Screen ON/OFF to save power B=0 or 1, 0 off, 1 on		For GLCD/OLED only	✓	✓	✓
DCB	D isplay C onfig on/off, the factory default set is on, so, when the module is powered up, it will display current communication mode on LCD, after you design finished, you can turn it off	displayConfig(0); displayConfig(1);	B=0 off, B=1 on	✓	✓	✓
SBB...	Set UART B aud, B are ASCII characters, the available values are: "300", "1200", "2400", "4800", "9600", "14400", "19200", "28800", "38400", "57600", "115200"	Set BAUD when initial the class	When adapter power up or reset, always start with 9600bps Baud rate	✓	✓	✓
SI2CAB	Set I2C Address, the default address is 0x27, the adapter will store the new address in memory	setI2CAddress(0x34);	Change address to 0x34	✓	✓	✓
STCRBB BBBB	Set T ext C olumns and R ows, this command will config your LCD if other than 1602 and the chip is other than KS0066U/F / HD44780	setLCDColRow(20,4);	The last 4 B should be "\x80\xC0\x94\xD4", it mapped the starting RAM address on LCD	✓		
TPBB	set T ext P osition for following display, BB are x and y	setPrintPos(x,y);	Only affect the following "TT" command	✓	✓	✓
TTB...	display TexT string, the text will wrapped in next row if the current row fulfilled, the Text Position will be changed to the last char dispalyed, this command ended by 0x00 or 0x0D received	print(string); print(int); print(char); print(float); print(double); drawStr(x,y,string);	The print function in Arduino, can also print other data and format the out put.	✓	✓	✓
MCDB	M anual C ommand: send command B to display bypass the adapter	directCommand(0xaf);	Use it if you want to control the display directly	✓	✓	✓
MDTB	M anual D aTa: send data B to display bypass the adapter	directData(0x88);	Same as above	✓	✓	✓

Graph Display Command: (B-one byte, B...-Bytes) for mono and color serial Graphic display only

Command	Description	Arduino lib function	note	Mono Graphic	Color Graphic
GPBB	set Graphic Position for following draw line command, BB are x and y in byte	setPrintPos(x,y,1);	X,y=0 to 255	✓	✓
DMB	Set the Display Mode for on coming command, the available values for B are: “!””~” not, “ ” or, “^” xor, “&” and, this means the next drawing pixel will logic operation with pixel already on screen.	setMode('!');	Like the Bitwise Operator in C	✓	✓
DIMBBB BB...	Display image width, 4 Each byte present 8 pixels, if the image width not divide 8, the last byte of a row only contain few pixels, eg. For width of 9 to 16, you need 2 bytes for a row	drawBitmap(x,y,width,height,*data);		✓	✓
SDB	Send graphic fuction Direction, the value of B is 0 to 3, represent0 to 270 degree respectively.	setRotation(0); undoRotation(); setRot90(); setRot180(); setRot270();	The setRotation(); will accept 0 to 3 represent0 to 270 degree respectively	✓	✓
CTB	Set display ConTrast, only for some models, Only for ST7565 LCD Controller	setContrast(30);	Only for ST7565 LCD Controller	✓	
FRBBBB	Draw a Filled Rectangle, 4 B are: X,Y(left top), X,Y (right bottom)	drawBox(x,y,width,height);	In order to compatible with u8g, drawBox() in Arduino use width and height	✓	✓
DRBBBB	Draw a Rectangle, 4 B are: X,Y(left top), X,Y (right bottom)	drawFrame(x,y,width,height);	drawFrame () in Arduino use width and height	✓	✓
CCBBBB	Draw a CirCle, 4 B are: X,Y, radius, filled or not	drawCircle(x,y,r,f); drawDisc(x,y,r);	f=1 means filled circel	✓	✓
DPBB	Draw a Pixel, 2 B are: x,y. The color was set up by commands of “SC” or “ESC”	drawPixel(x,y);		✓	✓
LNBBBB	Draw a Line from (x,y) to (x1,y1), 4 B are: x,y,x1,y1	drawLine(x,y,x1,y1); drawHLine(x,y,width); drawVLine(x,y,height);	drawHLine()-horizontal line drawVLine()-veritcal line	✓	✓
LTBB	Draw a Line from Tast position to (x,y), 2 B are:x,y	drawLineTo(x,y);		✓	✓
TRT	Move text cursor to next line(call Text ReTurn)	nextTextLine();	The y pixels moved depending on the font size current using	✓	✓
SFB	Set Font, follow by the font number, preloaded font number is: 6,10,18,51,120,123,0(default), user font number is 200,201,202,203 maps to 4 user font memory sections, you can combine adjacent sections together is the font size >4kb(each section has 4kb in size)	setFont(0);		✓	✓

SCB	Set Color for following display, this command affect all following drawing command, such as: text, line, circle, pixel, rectangle...	setColor(1);	0 and 1 for black white screen 0 to 255 for color screen	✓	✓
MABBB BBB	Move rectangle Area on screen to another place, the 6 B are represent: (x,y)(left- top),(w,h)(width-height), (xoffset,yoffset).	moveArea(x,y,w,h,xoffset,yoffset);		✓	✓
ETB	Enhanced set the current Text position Back to last char, this function will allow you display multiple chars at same position.	setTextPosBack();		✓	✓
ETOBB	Enhanced set Text position Offset, the 2 B are xoffset then yoffset, it will adjust the text position in pixels	setTextPosOffset(xoffset, yoffset);	0 to 255	✓	✓
ETPB	Enhanced set Text Position as pixels on screen, the 2 B are x, y coordinate on screen	setTextPosAbs(x,y);	X,y=0 to 255	✓	✓
SSSBBB ...	Set Start length, 2 nd following by data. For mono display, the start screen is bitmap data; For color display, the start screen is commands set, the first and second bytes are the commands length.	uploadStartScreen(1024, *data);	For mono display, the length of data should be: screen Width*High/8, eg. For 128x64 LCD, the length is 1024	✓	✓
SUFBBB B...	Set User upload, 2 nd the higher byte of data length, following by data	uploadUserFont(1,1434, *data);		✓	✓
DSSB	Display Start Screen stored in memory, also set up Automatic start screen display or not on next power up	displayStartScreen(1 or 0)	1= on, 0=off	✓	✓
DOUTB	Send a Byte to output head on board, the current driving ability for each pin is: 25mA (Sink/Source)	digitalOutput(0x1F);	The output head are vary from adapters	✓	✓
SLPB	Set Line Pattern when drawing line, only for new version firmware later than Jan. 2013. eg. B=0xAA is dot line, B=0xFA is dash line	setLinePattern(pattern);	Old version not support this fucntion	✓	✓
EDIM1B BBBB...	Enhanced Display Image, 1 byte color format. The following 4 bytes are start positon: x, y, image width, height, then following image data, each byte represent one pixel	drawBitmap256(x,y,width,height, *data)			✓
EDIM2B BBBB...	Enhanced Display Image, 2 byte color format. The following 4 bytes are start positon: x, y, image width, height, then following image data, each pixel occupy 2 bytes: B1:B0, the color structure: RRRRRGGG, GGGBBBBB, MSB first.	drawBitmap65K(x,y,width,height, *data)	Available on firmware version 2.7 and higher		✓
EDIM3B BBBB...	Enhanced Display Image, 3 byte color format. The following 4 bytes are start positon: x, y, image width, height, then following image data, 3 bytes represent one pixel: Red, Green, Blue, the validate value: 0 to 64 (6bits)	drawBitmap262K(x,y,width,height, *data)			✓
ESCB	Enhanced Set Color for following display, the BBB are color Red, color Green and Color Blue, this command affect all following drawing command, such as: text, line, circle, pixel, rectangle...	setTrueColor(R,G,B);			✓

Special Command: (B-one byte, B...-Bytes)

Command	Description	Arduino lib function	note
SLCDB	Only for multi-chip driver adapter: B=0 or '0' for ST7920 B=1 or '1' for KS0108 ("E" Low, "CS1"&"CS2" Low) B=2 or '2' for ST7565 Since product after Apr. 20 2013: B=3 or '3' for KS0108 ("E" Low, "CS1"&"CS2" High) B=4 or '4' for KS0108 , follow by effective level for "E", "CS1" and "CS2", eg. "SLCD4011" is same as "SLCD3"	setLCDChip(chip_num);	For Universal Graphic Serial LCD Adapter only

Set up universal graphic serial LCD adapter:

1) For ST7920 LCD controller:

- Verify and compare the pinout on adapter and LCD panel, they should be same on order (refer to the picture bellow);
- Pull up PSB pin (usually pin# 15) to VCC to let the LCD panel work at parallel mode;
- Connect the adapter with the LCD panel, adjust the contrast pot, power up the adapter, you should see the welcome screen displayed.

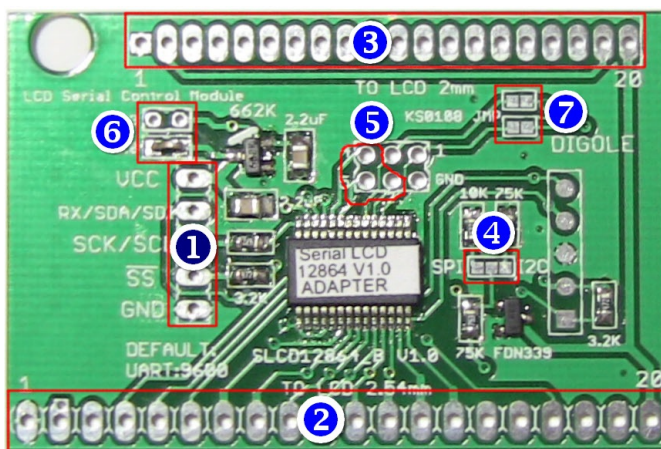
2) For KS0108 controller:

- Verify and compare the pinout on adapter and LCD panel, they should be same on order (refer to the picture bellow);
- Shorting the 2 KS0108 jumper: ⑦ on the picture;
- In most case you need add a 10K contrast pot by yourself;
- Connect the adapter with the LCD panel, power up the adapter, send chip configuration command (refer to above table) to adapter;
- Adjust the contrast pot, you should see the welcome screen displayed, if not, you probably need to try other value (1,3,4) in chip configuration command.

3) For ST7565 controller:

NOTE: There are vary of ST7565 LCD panels on market, and the pin out are vary from one module to others, this adapter only support SPI mode on the LCD panel, you usually need to connect pins on this adapter with LCD panel corresponding.

- Connect the pins on the adapter with LCD panel corresponding;
- Power up the adapter, then send chip configuration command to adapter ("SLCD2"), then you will see the welcome screen on the panel;
- You can use the set contrast command to adjust the contrast (ST7565 support software contrast adjustment).



① Communication port to MCU

② To LCD with 2.54mm

③ To LCD with 2.0mm

④ UART/I2C/SPI select jumper

Jumper:	I2c	SPI
UART	Open	Open
I2C	Short	Open
SPI	Open	Short

⑤ Digital out put @25mA

⑥ 5V, 3.3V Power Supply to LCD jumper,

DON'T SHORT on BOTH

⑦ KS0108 Jumper

Short both for KS0108

PIN CONFIGURATION																			
ST7920	Vss	Vdd	V0	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	PSB	NC	RST	Vout	BLA
KS0108	GND	Vdd	V0	D/I	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	CS1	CS2	RST	Vout	A+
ST7565	Vss	Vdd			D/C		CS	SCK	SID								RST	A+	K-

Pinout of this module connect to MCU:

PI N	Description	PI N	Description
1	GND (0V)	2	SS: SPI mode only chip select control in, low active
3	I2C and SPI mode: SCK/SCL: Clock in	4	UART mode: RX I2C mode: SDA SPI mode: SDI
5	VCC: power supply 1.8V to 5.5V depends on you LCD		

Connect with your master circuit:

