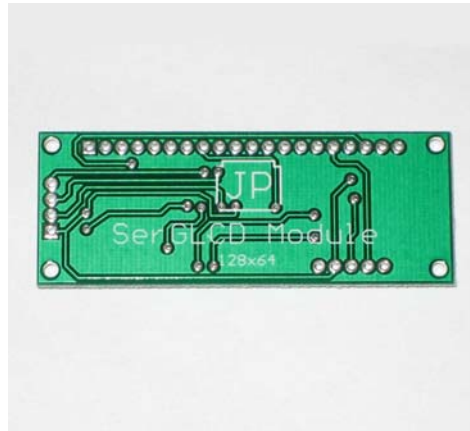
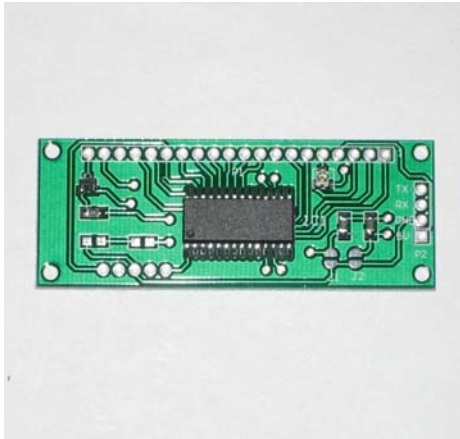


JP Serial GLCD 128 x 64 Module



JP serial GLCD 128x64 module is a simple and cost effective interface controller module. It supports Graphic LCD display base on KS0108/7 controller. Also it will work with any microcontroller. It has 6 font lib, 2 CGRam(1024Bytes), 12 CGRam(128Bytes), 5 text string Ram(18Bytes) and more useful functions. Also it will work with any microcontroller capable of accepting asynchronous serial data. (CGRam = Custom Image and Character Generator Ram)

SPECS

Power: 5V VDC <25mA (not include GLCD display)
Size: 65mm x 25mm (2.1" x 1.0")
Speed: 2400, 4800, 9600 and 19200 Baud.

PIN FUNCTIONS

P1 Connector: P1 connector will be connected to GLCD.

Pin1:	CS1
Pin2:	CS2
Pin3:	GND
Pin4:	Vcc
Pin5:	Vo
Pin6:	RS
Pin7:	R/W
Pin8:	E
Pin9 – Pin16:	D0 – D7
Pin17:	RST
Pin18:	Vee
Pin19:	LED+
Pin20:	LED

P2 Connector: P2 connector will be connected to power and a MCU

Rx: Serial input connection to JP Module. The module allows 2400, 4800, 9600 or 19200 baud based on J1 and J2 settings.

Tx: Serial output connection to JP Module. The module allows 2400, 4800, 9600 or 19200 baud based on J1 and J2 settings.

J1 and J2 Settings:

J1	J2	Baud
x	x	2400
–	x	4800
x	–	9600
–	–	19200

– = connected

x = disconnected

GND: Power supply and serial ground. This **MUST** also be connected to ground on the device to allow the serial data to be sent to the module.

5V: Supply voltage to module. Vin may be 5.0V (±0.5V), with 25 milliamps of current. (Not include GLCD Display)

JP Serial LCD Module Commands: (Command + Parameters + 250)

Those commands are ASCII value between 30 and 256. It needs a pause to be pressed.

Command (Hex)	Command (Dec)	Description	Parameters
\$21	33	Turn backlight On/Off	0 or 1
\$22	34	Turn Screen On/Off	0 or 1
\$23	35	Fill display with pattern	0 or 1
\$24	36	Select Font Size	0 - 6
\$25	37	Shifts display in Vertical direction	0 -63
\$26	38	Shifts a region to left or right	x, y, w, h, L or R
\$27	39	Invert a region	none
\$28	40	Sets the clip rectangle border	Left, Top, Right, Bottom
\$29	41	Turn Clipped On/Off	0 - 1
\$2A	42	Input Text Strings	none
\$2B	43	Input Text Strings to Ram	Text String Ram Number 0 - 4
\$31	49	Load internal small images 0 – 9,	none

\$32	50	Load small images to Ram (<=128B)	Size, Ram number (0-11)
\$33	51	Display small image from Ram(<=128)	x, y, w, h, ram number (0-11)
\$41	65	Load full size image to Ram 0	none
\$42	66	Load full size image to Ram1	none
\$43	67	Display full size image from Ram 0	none
\$44	68	Display full size image from Ram 1	none
\$45	69	Copy Screen to Ram 0	none
\$46	70	Copy Screen to Ram 1	none
\$47	71	Display copy screen data from Ram 0	none
\$48	72	Display copy screen data from Ram 1	none
\$49	73	Output Ram 0 data (1024 byte)	none
\$4A	74	Output Ram 1 data (1024 byte)	none
\$4B	75	Load Image to Ram 0 (<=1024 byte)	Size (word)
\$4C	76	Load Image to Ram 1 (<=1024 byte)	Size (word)
\$4D	77	Display image from ram 0 (<=1024 B)	x, y, w, h, dstyle*
\$4E	78	Display image from ram 1 (<=1024 B)	x, y, w, h, dstyle*
\$51	81	Draw a pixel	x, y, dstyle*
\$52	82	Draw a Horizontal line	x1, x2, y1, dstyle*
\$53	83	Draw a vertical line	x, y1, y,2 dstyle*
\$54	84	Draw a line	x1, y1 ,x2, y2, dstyle* , pstyle*
\$55	85	Draw a square	x, y, L, dstyle*
\$56	86	Draw a Rectangle	x1, y1, x2, y2, dstyle*
\$57	87	Draw a Round corner Rectangle	x1, y1, x2, y2, dstyle*
\$58	88	Draw a Ellipse	Cx, Cy, xRad, yRad, dstyle ,pstyle*
\$59	89	Draw a Circle	Cx, Cy, Radius, dstyle ,pstyle*
\$5A	90	Draw a solid box	x1, y1, x2, y2, dstyle*

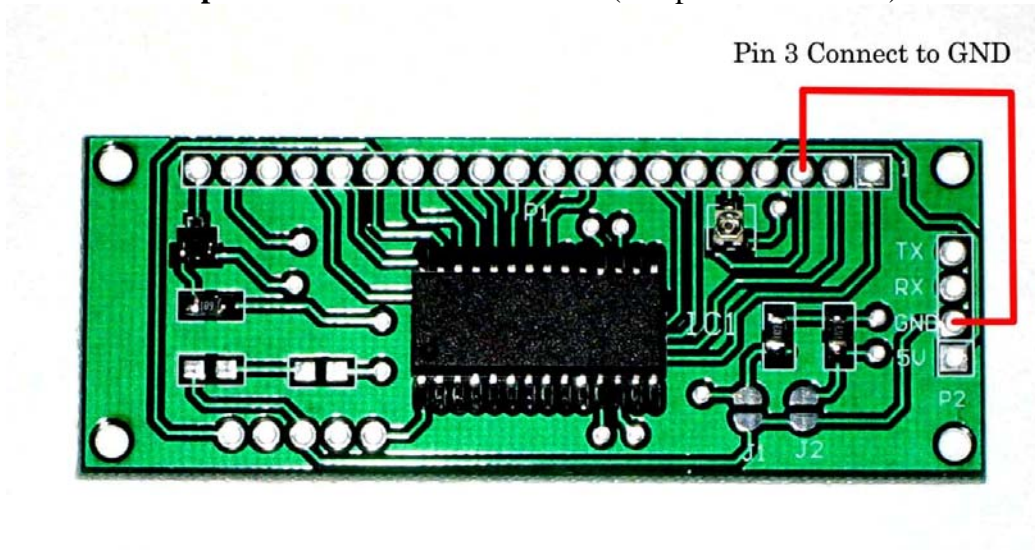
\$5B	91	Write a Char	Char, x, y, dstyle*
\$5E	92	Write text string to screen	x, y, dstyle*
\$5F	93	Write text string to screen from Ram	x, y, dstyle*, StringRam # (0 - 4)

* dstyle = Display Style, 0 = Erase, 1= write.

* pstyle = plotting styles, 0 = normal, 1 = dot, 2 = dash, 3 = longdash.

Note:

The module's pin3 have to connect to GND. (See picture for detail)



SERIAL DATA FORMAT

The serial data format is eight data bits, no parity and 1 stop bit (8N1). Characters are sent using standard ASCII values. Baud rate may be 2400, 4800, 9600 or 19200 depending on the settings of J1 and J2.

LIABILITY WARNING

This device should be used only for experimental purposes. It has NOT gone through extensive testing. You assume to take your own risk when you purchase this device, and release the responsibility and liability from the manufacturer with no harm.

REGULATORY WARNING

This device is intended solely for experimental purpose, it is not in finished product form and is NOT FCC approved. If you wish to install these modules into non-experimental final finished products, you will be responsible to have the modules approved by the FCC at your own cost.

Jianping Electronics
jp@jianpingusa.com
www.jianpingusa.com

JP SerGLCD 128x64 Module Test Code:

```
'=====
' File.....JP Serial GLCD 128x64 Module Test.BSP
' Purpose.....This test code use JP SerGLCD Module and BS2P
'           test Function.
' Auther.....Jianping Sun
' Email.....JP@JianpingUSA.com
' WebSite....www.JianpingUSA.com
' Started....Feb 26, 2010
' Updated.....
'=====
' {$STAMP BS2p}
' {$PBASIC 2.5}

SOUT          PIN      0           ' Serial output for GLCD
S_IN          PIN      1           ' Serial input  for GLCD
idx           VAR      Word        ' For loop
x             VAR      Byte

#SELECT $STAMP
#CASE BS2, BS2E, BS2PE
    T1200      CON      813
    T2400      CON      396
    T4800      CON      188
    T9600      CON      84
    T19K2      CON      32
    T38K4      CON      6
#CASE BS2SX, BS2P
    T1200      CON      2063
    T2400      CON      1021
    T4800      CON      500
    T9600      CON      240
    T19K2      CON      110
    T38K4      CON      45
#CASE BS2PX
    T1200      CON      3313
    T2400      CON      1646
    T480       CON      813
    T9600      CON      396
    T19K2      CON      188
```

```
T38K4      CON      84
#ENDSELECT
```

```
Inverted   CON      $4000
Baud       CON      T19K2 '+ Inverted
```

```
'=====
'                                     Main program
'=====
```

```
Main:
```

```
    PAUSE 1500
    SEROUT SOUT, Baud, [$61]    ' Send Command to show JP Logo
    SEROUT SOUT, Baud, [250]
    PAUSE 1000
    x = 3                      'Set font #3
    GOSUB SetFont
    GOSUB CleanScreen
    GOSUB BackLightTest
    GOSUB CleanScreen
    GOSUB ScreenOnOffTest
    GOSUB CleanScreen
    GOSUB SetLine
    GOSUB CleanScreen
    GOSUB FunctionTest
    GOSUB CleanScreen
```

```
END
```

```
'===== Ellipse, Circle, Rectang, Square, Box Test =====
```

```
FunctionTest:
```

```
    SEROUT SOUT, Baud, [$58]    'Ellipse function
    SEROUT SOUT, Baud, [64]     'x1
    SEROUT SOUT, Baud, [32]     'y1
    SEROUT SOUT, Baud, [60]     'x2
    SEROUT SOUT, Baud, [30]     'y2
    SEROUT SOUT, Baud, [1]
    SEROUT SOUT, Baud, [0]
    SEROUT SOUT, Baud, [250]
    PAUSE 1000
```

```
    SEROUT SOUT, Baud, [$28]    'ClipRect function
    SEROUT SOUT, Baud, [5]      'Left
    SEROUT SOUT, Baud, [5]      'Top
    SEROUT SOUT, Baud, [122]    'Right
    SEROUT SOUT, Baud, [58]     'Bottom
    SEROUT SOUT, Baud, [250]
    PAUSE 100
```

```
    SEROUT SOUT, Baud, [$29]    'ClipPed function
    SEROUT SOUT, Baud, [0]      ' true
    SEROUT SOUT, Baud, [250]
    PAUSE 100
```

```
FOR x = 0 TO 6
```

```
    SEROUT SOUT, Baud, [$59]    'Circle function
    SEROUT SOUT, Baud, [64]     'x
    SEROUT SOUT, Baud, [32]     'y
    SEROUT SOUT, Baud, [8*x+2]  'r
    SEROUT SOUT, Baud, [1]
```

```

SEROUT SOUT, Baud, [0]
SEROUT SOUT, Baud, [250]
PAUSE 1000
NEXT

SEROUT SOUT, Baud, [$29]      'ClipPed function
SEROUT SOUT, Baud, [1]       'false
SEROUT SOUT, Baud, [250]
PAUSE 100

SEROUT SOUT, Baud, [$56]      'Rectang function
SEROUT SOUT, Baud, [8]       'x1
SEROUT SOUT, Baud, [8]       'y1
SEROUT SOUT, Baud, [120]     'x2
SEROUT SOUT, Baud, [56]     'y2
SEROUT SOUT, Baud, [1]
SEROUT SOUT, Baud, [250]
PAUSE 2000

SEROUT SOUT, Baud, [$57]      'Round Rectang function
SEROUT SOUT, Baud, [12]     'x1
SEROUT SOUT, Baud, [12]     'y1
SEROUT SOUT, Baud, [116]    'x2
SEROUT SOUT, Baud, [52]     'y2
SEROUT SOUT, Baud, [1]
SEROUT SOUT, Baud, [250]
PAUSE 2000

SEROUT SOUT, Baud, [$55]      'Square function
SEROUT SOUT, Baud, [48]     'x1
SEROUT SOUT, Baud, [16]     'y1
SEROUT SOUT, Baud, [33]     'L
SEROUT SOUT, Baud, [1]
SEROUT SOUT, Baud, [250]
PAUSE 1000

SEROUT SOUT, Baud, [$5A]      'Full box function
SEROUT SOUT, Baud, [51]     'x1
SEROUT SOUT, Baud, [19]     'y1
SEROUT SOUT, Baud, [78]     'x2
SEROUT SOUT, Baud, [46]     'y2
SEROUT SOUT, Baud, [1]
SEROUT SOUT, Baud, [250]
PAUSE 1000

FOR x = 0 TO 8
SEROUT SOUT, Baud, [$27]      'Invert function
SEROUT SOUT, Baud, [10]     'x1
SEROUT SOUT, Baud, [10]     'y1
SEROUT SOUT, Baud, [109]    'w
SEROUT SOUT, Baud, [45]     'h
SEROUT SOUT, Baud, [250]
PAUSE 1000
NEXT
PAUSE 3000
RETURN
'===== Line Test =====

```

```

SetLine:
  SEROUT SOUT, Baud, [$52]      'Line Hor function
  SEROUT SOUT, Baud, [0]       'x1
  SEROUT SOUT, Baud, [127]    'x2
  SEROUT SOUT, Baud, [31]     'y
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000

  SEROUT SOUT, Baud, [$53]    'Line Ver function
  SEROUT SOUT, Baud, [63]    'x1
  SEROUT SOUT, Baud, [0]     'y1
  SEROUT SOUT, Baud, [63]    'y2
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000

FOR x = 0 TO 5
  SEROUT SOUT, Baud, [$54]    'Line function
  SEROUT SOUT, Baud, [0]     'x1
  SEROUT SOUT, Baud, [x*10+5] 'y1
  SEROUT SOUT, Baud, [127]   'x2
  SEROUT SOUT, Baud, [x*10+5] 'y2
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [x]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000
NEXT

FOR x = 0 TO 3
  SEROUT SOUT, Baud, [$54]    'Line function
  SEROUT SOUT, Baud, [32*x+16] 'x1
  SEROUT SOUT, Baud, [0]     'y1
  SEROUT SOUT, Baud, [32*x+16] 'x2
  SEROUT SOUT, Baud, [63]    'y2
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [x]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000
NEXT

  SEROUT SOUT, Baud, [$54]    'Line function
  SEROUT SOUT, Baud, [0]     'x1
  SEROUT SOUT, Baud, [0]     'y1
  SEROUT SOUT, Baud, [127]   'x2
  SEROUT SOUT, Baud, [63]    'y2
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [0]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000

  SEROUT SOUT, Baud, [$54]    'Line function
  SEROUT SOUT, Baud, [0]     'x1
  SEROUT SOUT, Baud, [63]    'y1
  SEROUT SOUT, Baud, [127]   'x2
  SEROUT SOUT, Baud, [0]     'y2
  SEROUT SOUT, Baud, [1]

```

```

        SEROUT SOUT, Baud, [0]
        SEROUT SOUT, Baud, [250]
        PAUSE 1000
PAUSE 1000
RETURN
'===== Clean Screen =====
CleanScreen:
        SEROUT SOUT, Baud, [$23]
        SEROUT SOUT, Baud, [0]
        SEROUT SOUT, Baud, [250]
        PAUSE 100
RETURN
'===== Set Font =====
SetFont:
        SEROUT SOUT, Baud, [$24]
        SEROUT SOUT, Baud, [x]
        SEROUT SOUT, Baud, [250]
        PAUSE 100
RETURN
'===== Back Light Test =====
ScreenOnOffTest:
        GOSUB ScreenText1
        PAUSE 1000
        x = 1
        GOSUB ScreenOnOff
        x = 0
        GOSUB ScreenOnOff
        x = 1
        GOSUB ScreenOnOff
        x = 0
        GOSUB ScreenOnOff
RETURN
'===== Back Light =====
ScreenOnOff:
        SEROUT SOUT, Baud, [$22]
        SEROUT SOUT, Baud, [x]
        SEROUT SOUT, Baud, [250]
        PAUSE 1000
RETURN
'===== Screen Text =====
ScreenText1:
        SEROUT SOUT, Baud, [$2B]
        SEROUT SOUT, Baud, [1]
        SEROUT SOUT, Baud, ["Screen On/Off"]
        SEROUT SOUT, Baud, [250]
        PAUSE 100
        SEROUT SOUT, Baud, [$5F]
        SEROUT SOUT, Baud, [5]
        SEROUT SOUT, Baud, [25]
        SEROUT SOUT, Baud, [1]
        SEROUT SOUT, Baud, [1]
        SEROUT SOUT, Baud, [250]
        PAUSE 1000
RETURN
'===== Back Light Test =====
BackLightTest:
        GOSUB ScreenText0

```

```

x = 0
GOSUB BackLight
x = 1
GOSUB BackLight
x = 0
GOSUB BackLight
x = 1
GOSUB BackLight
RETURN
'===== Back Light =====
BackLight:
  SEROUT SOUT, Baud, [$21]
  SEROUT SOUT, Baud, [x]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000
RETURN
'===== Screen Text =====
ScreenText0:
  SEROUT SOUT, Baud, [$2B]
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, ["BackLight Test."]
  SEROUT SOUT, Baud, [250]
  PAUSE 100
  SEROUT SOUT, Baud, [$5F]
  SEROUT SOUT, Baud, [0]
  SEROUT SOUT, Baud, [25]
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [1]
  SEROUT SOUT, Baud, [250]
  PAUSE 1000
RETURN

```