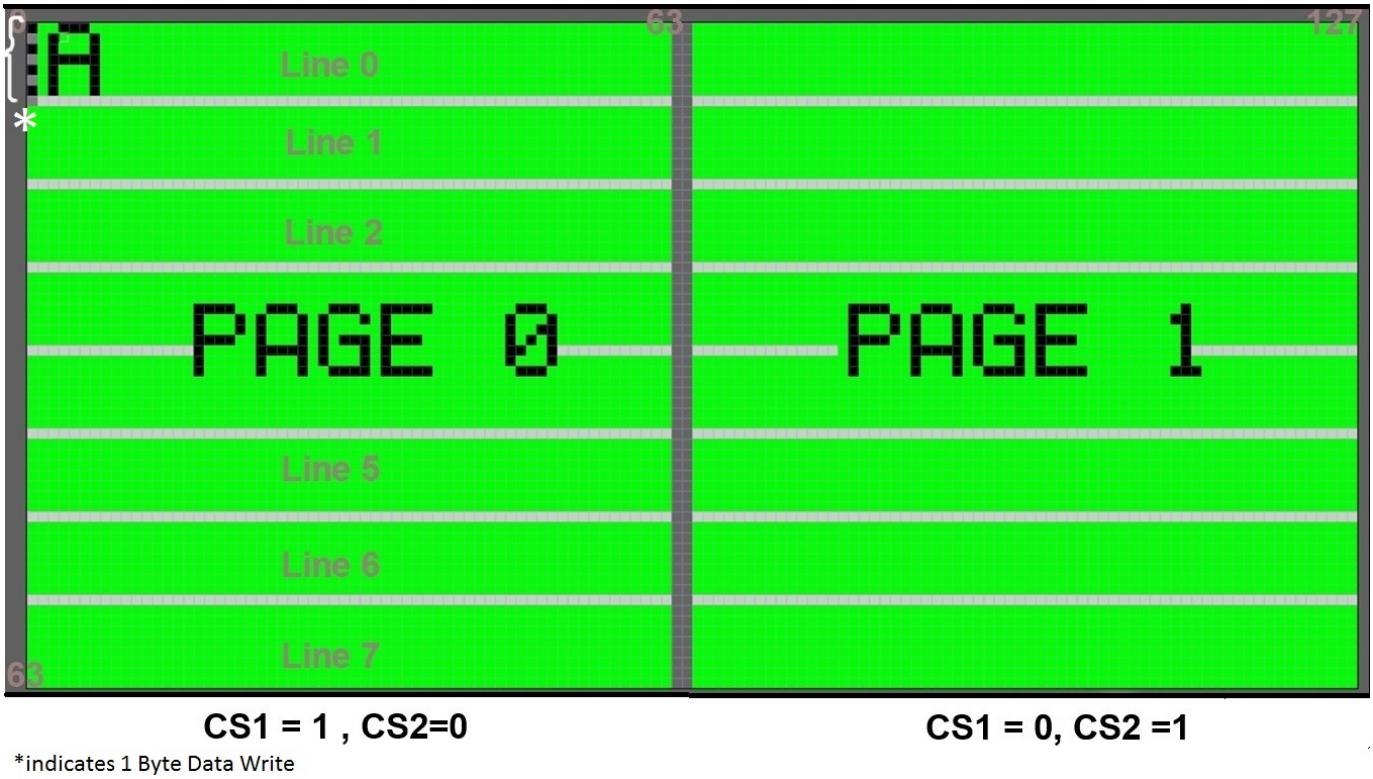
Interfacing KS0108 based JHD12864E Graphics LCD with Atmega32

Contents

(/wiki/File:DSC01793.JPG)

In this tutorial we will look at interfacing KS0108 display controller based JHD12864E display. There are many displays out there based on KS0108 or compatible display controller. They all work the same way, but make sure to check the datasheet for the pin diagram because the pin layout is not uniform, if you've a display from different manufacturer. We will look at the working of the display, the hardware setup and programming with ATmega32. You may use any other AVR MCU as well. You may port the library for other MCUs as well. We have it tested and working on 8051 PIC and ARM. We may do a similar tutorial on these MCUs as well. Unlike a 16 x 2 display, this does not have a character map for ascii values stored on its ROM. However it allows us the flexibility of creating fonts like Arial, times new roman etc. We could also display bit-map images on it and stretching it little further we can make GUI's and little animation, but that's for another day. So lets get started.

Working of KS0108 Display Controller



(/wiki/File:Pixel_arrangement.JPG)

The image tells the story of the a 128 x 64 display. The display that we used for the tutorial is JHD12864E and it has two KS0108 controllers in built. Each controller controls 64 x 64 pixels. Hence the display is 128 pixels wide and 64 pixels in height. Each pixel is either ON(black) or OFF(green), a monochrome display indeed.

- The entire display is divided into 2 pages.
- Each page has 64 x 64 pixels.
- Each page in-turn is divided into 8 lines.
- Each line has I do not know what to say. It has 64 columns and each column is 8 pixel tall. We have decided to call it the cursor position.
- Now when you write a byte to the LCD from the micro-controller it is displayed on that column as marked by * in the image.

If you've read and followed what is written above, make the image below full screen and observe it carefully. It has taken me an hour to make that image.

Instruction Set

If you've used a 16x2 displayed, this works exactly like it. However note that there is no ROM on this LCD. Hence the ASCII character patterns are not stored any where. We can make character fonts as we wish. If you download the code, you'll see we have implemented all the characters with 5x8 pixel font.

- Command Write: Used to set cursor line positions and set up the display.
- Data Write: Writes a byte on the display as discussed earlier.

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function
Display on/off	L	L	L	L	H	H	H	H	H	L/H	Controls the display on or off. Internal status and display RAM data is not affected. L: OFF, H: ON
Set Cursor (Y address)	L	L	L	H	Y address (0 - 63)						Sets the Y address in the Y address counter.
Set Line (X address)	L	L	H	L	H	H	H	Page (0 - 7)			Sets the X address at the X address register.
Display start line (Z address)	L	L	H	H	Display start line (0 - 63)						Indicates the display data RAM displayed at the top of the screen.
Status read	L	H	Busy	L	On/ Off	Reset	L	L	L	L	Read status. BUSY L: Ready H: In operation ON/OFF L: Display ON H: Display OFF RESET L: Normal H: Reset
Write display data	H	L	Write data								Writes data (DB0:7) into display data RAM. After writing instruction, Y address is increased by 1 automatically.
Read display data	H	H	Read data								Reads data (DB0:7) from display data RAM to the data bus.

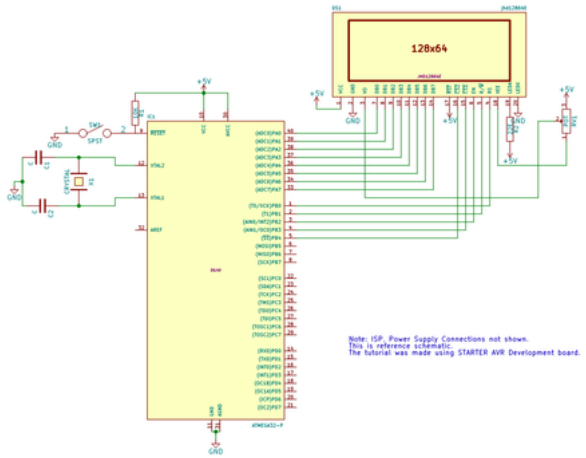
Do not worry about this

* Datasheet modified to help you understand :)

(/wiki/File:Instruction_Set.JPG)

Hardware Setup

Schematic Diagram and Pin Description



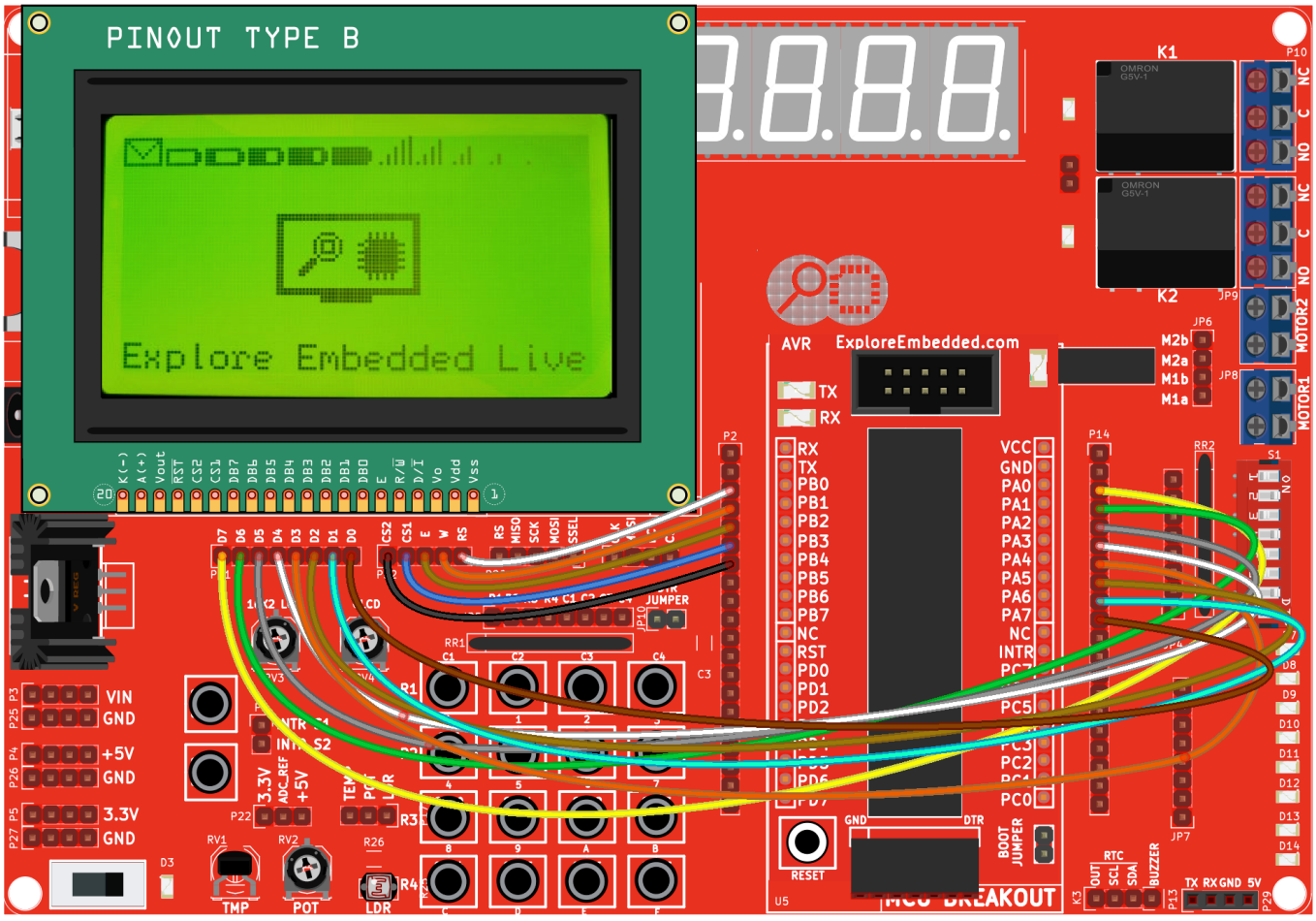
(/wiki/File:GLCD_Atmega32.png)

PIN NO.	SYMBOL	DESCRIPTION	FUNCTION
1	VSS	GROUND	0V (GND)
2	VDD	POWER SUPPLY FOR LOGIC CIRCUIT	+5V
3	V0	LCD CONTRAST ADJUSTMENT	
4	RS	INSTRUCTION/DATA REGISTER SELECTION	RS = 0 : INSTRUCTION REGISTER RS = 1 : DATA REGISTER
5	R/W	READ/WRITE SELECTION	R/W = 0 : REGISTER WRITE R/W = 1 : REGISTER READ
6	E	ENABLE SIGNAL	
7	DB0	DATA INPUT/OUTPUT LINES	8 BIT: DB0-DB7
8	DB1		
9	DB2		
10	DB3		
11	DB4		
12	DB5		
13	DB6		
14	DB7		
15	CS1	CHIP SELECTION	CS1=1,CHIP SELECT SIGNAL FOR IC1
16	CS2	CHIP SELECTION	CS2=1,CHIP SELECT SIGNAL FOR IC2
17	RST	RESET SIGNAL	RSTB=0,DISPLAY OFF,DISPLAY FROM LINE 0.
18	VEE	NEGATIVE VOLTAGE FOR LCD DRIVING	-10V
19	LED+	SUPPLY VOLTAGE FOR LED+	+5V
20	LED-	SUPPLY VOLTAGE FOR LED-	0V

(/wiki/File:Pin_Definition_JHD12864E.JPG)

I know all the pins are self explanatory. However not the pin number 18. It generates negative voltage required for the display. A 10K ohm pot is connected as shown in the circuit to generate -10V.

Hookup



(/wiki/File:KS0108_JHD12864E_LCD_with_Atmega32.png)

Code

To display text

```
1  #include "glcd.h"       //User defined LCD library which conatins the lcd routines
```

```
/* start the main program */
void main()
{
        GLCD_Init();
        GLCD_Printf("Hello World!");
        GLCD_GoToLine(4);
        GLCD_Printf("*&^%$#@!~");
        GLCD_GoToLine(7);
        GLCD_DisplayString("Well this is the end!");
        while(1);
 }
```

glcd_atmega32.c view raw (https://gist.github.com/Xplorer001/23d3c552154a3b5776a4/raw/923c66fe9df19eb7eb02149dd9badc6b811ff9d2/glcd_atmega32.c) (https://gist.github.com/Xplorer001/23d3c552154a3b5776a4#file-glcd_atmega32-c) hosted with ♥ by GitHub (https://github.com)

(/wiki/File:DSC01796.JPG)

The output


(/wiki/File:DSC01795.JPG)

The HW Setup

To display bitmap image

Download

Head over to Github to download all that you need (https://github.com/ExploreEmbedded/Tutorials/tree/master/AVR%20Tutorials/Interfacing%20GLCD%20JHD12864E)

Well there is lot more to do with the GLCD, and we will cover it in the future tutorials. I am looking forward for your comments, doubts, suggestion or feedback. Or if you just want to stop by and say hello!

Category (/wiki/Special:Categories): AVR Tutorials (/wiki/Category:AVR_Tutorials)