PHOTO SLIDE SHOW ON GRAPHICS LCD USING ARDUINO ATMEGA 328 MICROCONTROLER

Varun Varadan

Abstract: There are many types of LCDs available in the market. The Graphics LCD is called Graphics because of its pixel nature. The lines LCDs only possess the ability to display the pixels in a rectangular area and pixels are discontinuous. In a GLCD, the pixels are spread all over. We can use it to display characters, graphics, as well as images. This paper aims to demonstrate the process of displaying graphical images on a graphics LCD using a simple Arduino Atmega 328 microcontroller board. The images to be displayed on the GLCD cannot be just loaded onto it directly. It has to be processed by a processing software which converts the image files(in .jpeg, .png, .bmp forms) into its equivalent bitmaps (i.e. representation of the jpeg image information in the form of binary 0s and 1s). These bitmaps are then uploaded to the GLCD which will then display the equivalent black and white output of the image. The images to be loaded onto the GLCD screen are first stored in the library of the Arduino.

Index Terms: Arduino, Atmega 328, Graphics LCD, Microcontroller, Photos

1 Introduction

A graphical LCD screen or GLCD (acronym of English Graphic Liquid Crystal Display) is a flat panel made up of a matrix of pixels monochrome placed in front of a light source or reflector. Often used batteries in electronic devices because it uses very small amounts of electricity, there are different versions of screens with embedded controllers such as the Samsung KS0107, Samsung KS0108 or the Toshiba T6963.

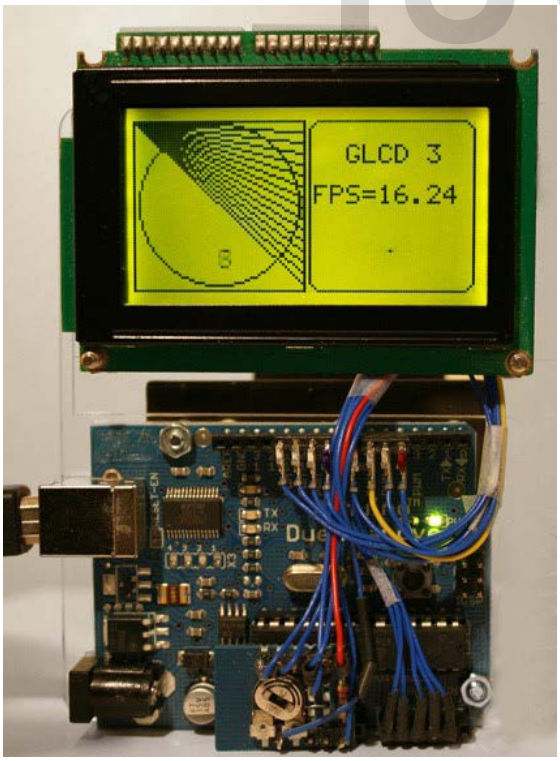

Fig 1: A typical GLCD

It has a RAM -size internal capacity available to the screen, for example if a screen has a size of 128 pixels long by 64 pixels tall (128x64) has a RAM inside of the same capacity (128x64). They are usually handled by microcontrollers for configuration and use of the same.

We have used an ARDUINO ATMEGA328 microcontroller board with the GLCD library to program and enable the display of pictures on the GLCD screen. This library not only enables us to display pictures or bitmaps, but also let us manipulate the output as we wanted.

We have also made use of software named PROCESSING for converting .jpeg, .png, .bmp images into bitmap codes in

HEX and later loaded this bitmap code into the microcontroller which in return displayed the bitmap image on the GLCD pixels. The bitmap HEX code enables or activates the said pixels to glow either black or white and this is displayed on the background which could also be white or black respectively.

2 Block diagram and Circuit diagram

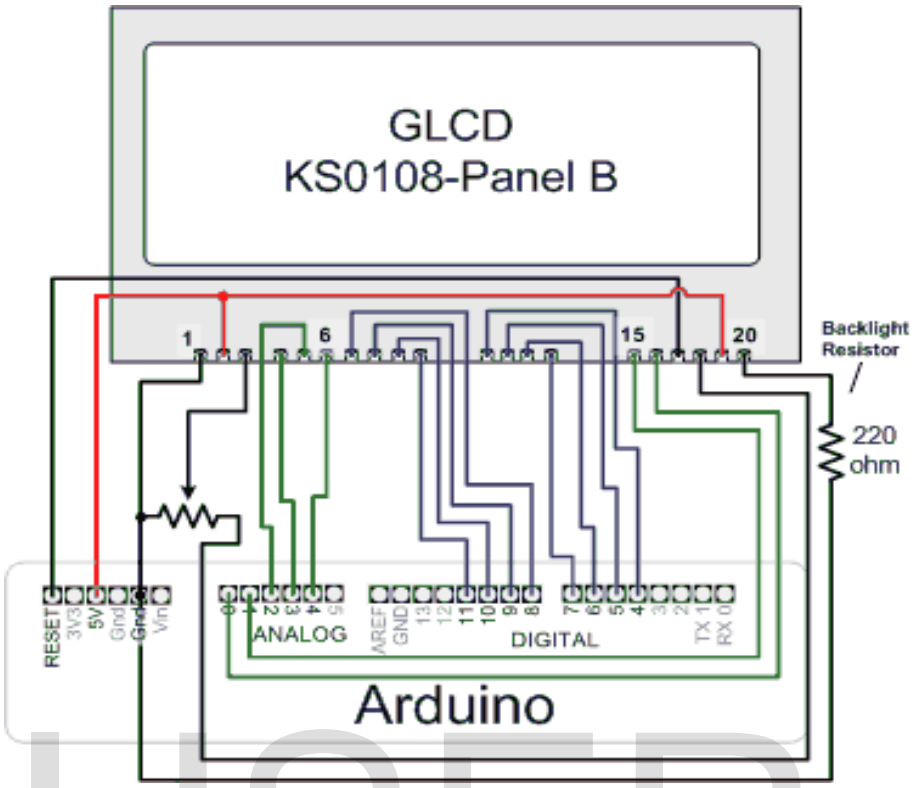

Fig 2: Circuit diagram of setup

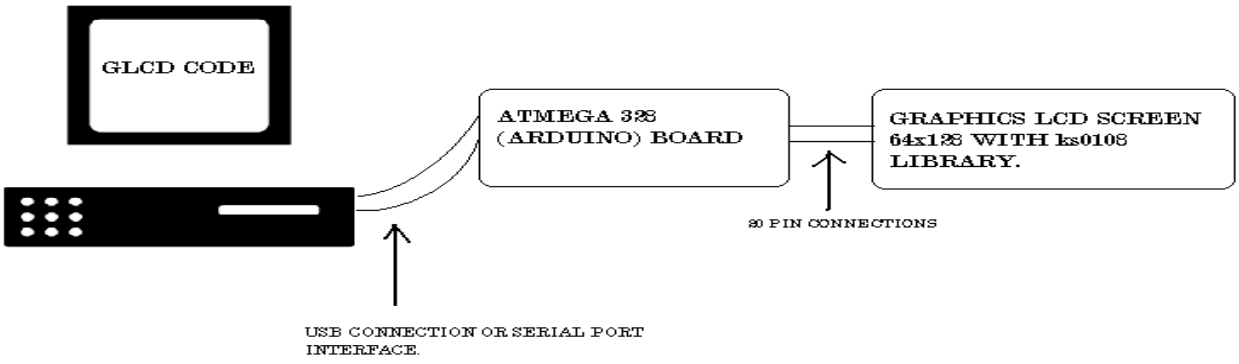

Fig 3: Block diagram of experiment

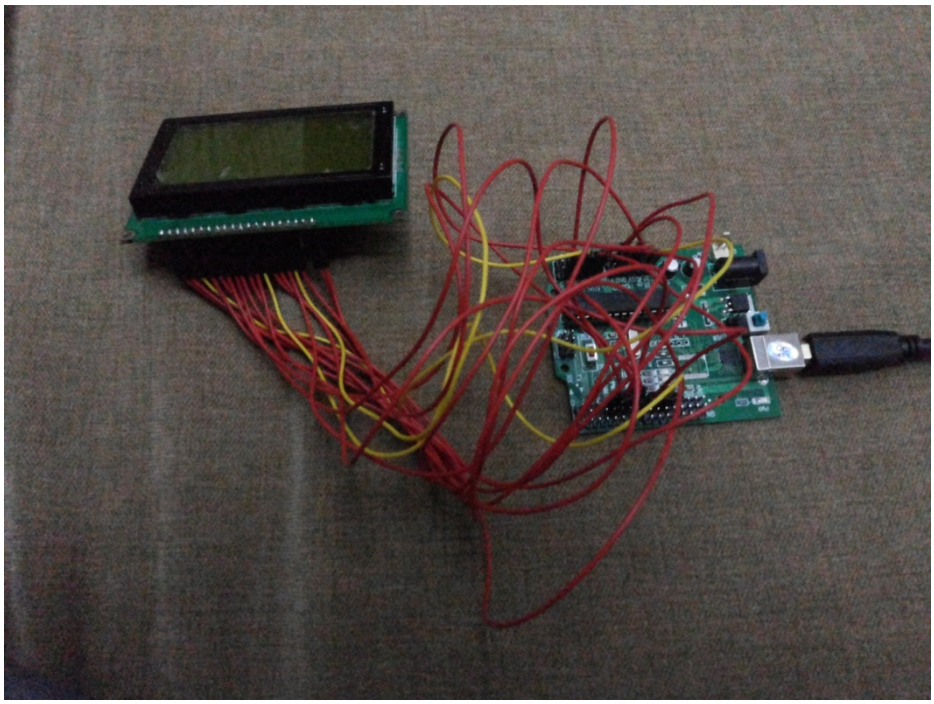

Fig 4: experimental setup

3 Working

The connections made from arduino board to the GLCD enable the flow of control and data signals from the microcontroller to the GLCD screen. Pins 1,3,18 and 20 are used for brightness/contrast control of the GLCD screen. The other connections are for power, control signals, and data transfer.

The bitmaps loaded into the microcontroller are transferred to the GLCD through these data lines (pins 7 to 14) and then loaded onto the screen. The code created for the display contains information of the bitmap size, its layout and its placement on the 64x128 screen and also its pixel configuration.

Once the image is displayed, the screen is cleared and another image is displayed and all these images are locked into a continuous display loop which goes on till the power is pulled.

4 Code used

First, the image files are loaded into the library of the Arduino. Then, the bitmap files are created using the processing software. The header files of these bitmaps are included in the Arduino code as shown below:

```
#include<glcd.h>

#include "bitmaps/butterfly.h"

#include "bitmaps/firefox.h"

#include "bitmaps/allBitmaps.h"

#include "bitmaps/halloween.h"

#include "bitmaps/ninja.h"

#include<fonts/allFonts.h>

 Image_t icon;

void setup()

{

  GLCD.Init();

}

void loop()

{

  GLCD.ClearScreen();

  delay(1000);
```

```
icon = butterfly;
GLCD.DrawBitmap(icon,64,0);
GLCD.CursorTo(0,0);
GLCD.print("Butterfly");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = ninja;
GLCD.DrawBitmap(icon,0,0);
GLCD.CursorTo(0,10);
GLCD.print("Ninja");
GLCD.CursorTo(0,11);
GLCD.print("Turtle");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = firefox;
GLCD.DrawBitmap(icon,64,0);
GLCD.CursorTo(0,0);
GLCD.print("Mozilla");
GLCD.CursorTo(0,1);
GLCD.print("FireFox");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = halloween;
GLCD.DrawBitmap(icon,0,0);
GLCD.CursorTo(0,10);
GLCD.print("Halloween");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = butterfly;
GLCD.DrawBitmap(icon,64,0,WHITE);
GLCD.CursorTo(0,0);
GLCD.print("Butterfly");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = ninja;
GLCD.DrawBitmap(icon,0,0,WHITE);
GLCD.CursorTo(0,10);
GLCD.print("Ninja");
GLCD.CursorTo(0,11);
GLCD.print("Turtle");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = firefox;
GLCD.DrawBitmap(icon,64,0,WHITE);
GLCD.CursorTo(0,0);
GLCD.print("Mozilla");
GLCD.CursorTo(0,1);
GLCD.print("FireFox");
delay(1000);
GLCD.ClearScreen();
delay(1000);
icon = halloween;
```

```
 GLCD.DrawBitmap(icon,0,0,WHITE);

 GLCD.CursorTo(0,10);

 GLCD.print("Halloween");

 delay(1000);

 GLCD.ClearScreen();

 delay(1000);

 GLCD.print("THANK YOU");

 delay(1000);

}
```

5 Output Stages

Butterfly with black on white background pixel configuration.

Ninja Turtle with black on white background pixel configuration.

Mozilla Firefox with black on white background pixel configuration.

Halloween with black on white background pixel configuration.

Butterfly with white on black (inverted) background pixel configuration.

Ninja Turtle with white on black (inverted) background pixel configuration.

Mozilla Firefox with white on black (inverted) background pixel configuration

Halloween with white on black (inverted) background pixel configuration.

End Screen

6 Advantages

Any image which fits onto the GLCD screen can be displayed on it using this code and principle. On working with such projects, one can get well acquainted with the use and operation of an Arduino, GLCD, and also coding it and the processing that works behind it.

7 Disadvantages

Images with a variety of colours do not get displayed properly, since all the information is converted into either 'black' or 'white' in a GLCD. Prior processing and inclusion in the library is required to display any new image. One cannot directly click a picture and automatically display it without the above process.

8 Conclusion and future scope

In this experiment we have worked with a GLCD and seen its versatility by using it as a tool for a photo slide show, using the help of processing and Arduino. The GLCD does have its few drawbacks as mentioned above. However, it can still be used to develop games. Screen savers can be designed on such working principles. It can be used as display boards at stations and airports, etc. Its ease of programming and use its its biggest plus point.

The photo-slide show using GLCD can be used for displaying images in different fields of life. For example, it can be used in GLCD-Touchpad based restaurant ordering and automatic service system; GLCD touchpad for phones and computers; display of random messages on the roads like 'drive safe' etc.