APPLICATION NOTE

****

4D SYSTEMS

*TURNING TECHNOLOGY INTO ART*

**Serial Arduino Block Image Transfer**

**Document Date: 1th September 2014**

**Document Revision: 1.0**

# Description

This Application Note explores the possibilities provided by the Serial environment in Workshop for a 4D display module to work with an Arduino host. In this example, the host is an Arduino Uno board. The host can also be an Arduino Mega 2560 or Due. Ideally, the application described in this document should work with any Arduino board that supports software serial or with at least one UART serial port. [See specifications of Aduino boards here.](http://arduino.cc/en/Products.Compare)

Before getting started, the following are required:

* Any of the following 4D Picaso display modules:

|  |  |  |
| --- | --- | --- |
| [uLCD-24PTU](http://www.4dsystems.com.au/product/1/7/4D_Intelligent_Display_Modules/uLCD_24PTU/) | [uLCD-32PTU](http://www.4dsystems.com.au/product/1/9/4D_Intelligent_Display_Modules/uLCD_32PTU/) | [uLCD-43(P/PT/PCT)](http://www.4dsystems.com.au/product/1/11/4D_Intelligent_Display_Modules/uLCD_43/) |
| [uLCD-28PTU](http://www.4dsystems.com.au/product/1/8/4D_Intelligent_Display_Modules/uLCD_28PTU/) | [uLCD-32WPTU](http://www.4dsystems.com.au/product/1/10/4D_Intelligent_Display_Modules/uLCD_32WPTU/) | [uVGA-III](http://www.4dsystems.com.au/product/1/124/4D_Intelligent_Display_Modules/uVGA_III/) |

and other superseded modules which support the ViSi Genie environment

* The target module can also be a Diablo16 display

|  |  |
| --- | --- |
| [uLCD-35DT](http://www.4dsystems.com.au/product/uLCD_35DT/) | [uLCD-70DT](http://www.4dsystems.com.au/product/uLCD_70DT/) |

Visit [www.4dsystems.com.au/products](http://www.4dsystems.com.au/products) to see the latest display module products that use the Diablo16 processor. The display module used in this application note is the uLCD-32PTU, which is a Picaso display. This application note is applicable to Diablo16 display modules as well.

* [4D Programming Cable](http://www.4dsystems.com.au/product/17/114/Accessories/4D-Programming-Cable/) or [µUSB-PA5](http://www.4dsystems.com.au/product/17/115/Accessories/uUSB-PA5/)
* [micro-SD (µSD)](http://www.4dsystems.com.au/product/uSD_2GB/) memory card
* [Workshop 4 IDE](http://www.4dsystems.com.au/product/10/120/Development/4D_Workshop_4_IDE/) (installed according to the installation document)
* Any Arduino board with a UART serial port
* microSD Card Shield for Arduino
* [4D Arduino Adaptor Shield](http://www.4dsystems.com.au/product/4D_Arduino_Adaptor_Shield_II/) (optional) or connecting wires
* [Arduino IDE](http://arduino.cc/en/Main/Software)
* When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented in these recommended application notes.

# Content

[Description 2](#_Toc397352798)

[Content 3](#_Toc397352799)

[Application Overview 3](#_Toc397352800)

[Setup Procedures 3](#_Toc397352801)

[Using the LCD Image Converter 3](#_Toc397352802)

[Program the Arduino Host 5](#_Toc397352803)

[Initialize SD shield 5](#_Toc397352804)

[Get Width and Height 6](#_Toc397352805)

[Perform BLIT 7](#_Toc397352806)

[Connect the 4D Display Module to the Arduino Host 8](#_Toc397352807)

[Proprietary Information 9](#_Toc397352808)

[Disclaimer of Warranties & Limitation of Liability 9](#_Toc397352809)

# Application Overview

This application note shows how to perform Block Image Transfer using the Arduino Serial library. The sample program in this application note does not use the Blit Com to Display command defined in the library. Instead, a new function to directly transfer pixel values to the display from the uSD card is defined and used because of flash memory limitations in some Arduino boards. A third party application is used to convert an image to a character array. The character array contains the hex value for each pixel of the image.

# Setup Procedures

The display must be configured as a slave device first before it can be controlled by a host. For instructions on how to launch Workshop 4, how to connect the display module to the PC, and how to configure the display as a slave device, kindly refer to the section “**Setup Procedure**” of any of the application notes below. Choose according to your display module’s processor.

[**Serial Goldelox Getting Started - The SPE Application**](http://www.4dsystems.com.au/appnote/4D-AN-00086/)

[**Serial Picaso Getting Started - The SPE Application**](http://www.4dsystems.com.au/appnote/4D-AN-00090/)

[**Serial Diablo16 Getting Started - The SPE Application**](http://www.4dsystems.com.au/appnote/4D-AN-00110/)

These application notes also introduce the user to the Serial Protocol thru the use of the Serial Commander.

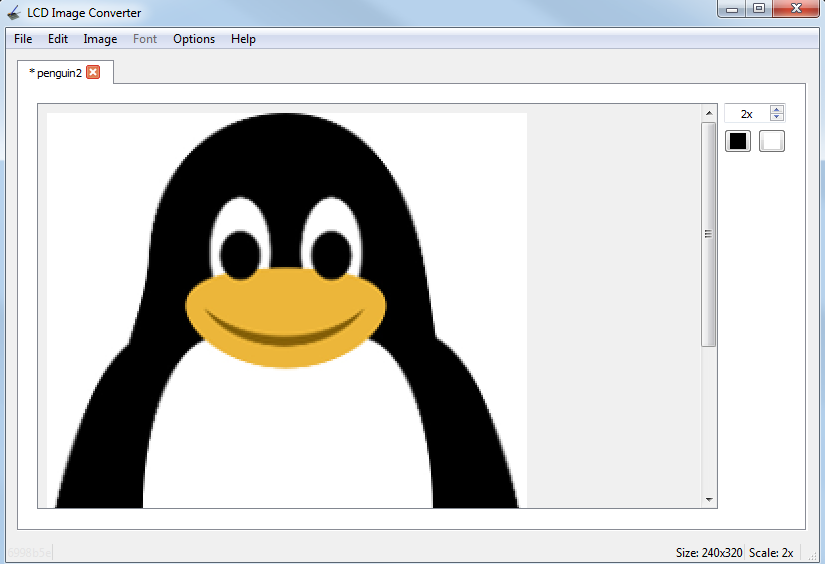
# Using the LCD Image Converter

A third party application is used to convert an image to a character array. The application can be downloaded here <https://code.google.com/p/lcd-image-converter/> . After the image is converted, the content of the array will be then copied to a text file. The text file has the following format to be followed:   
  
[width, height]

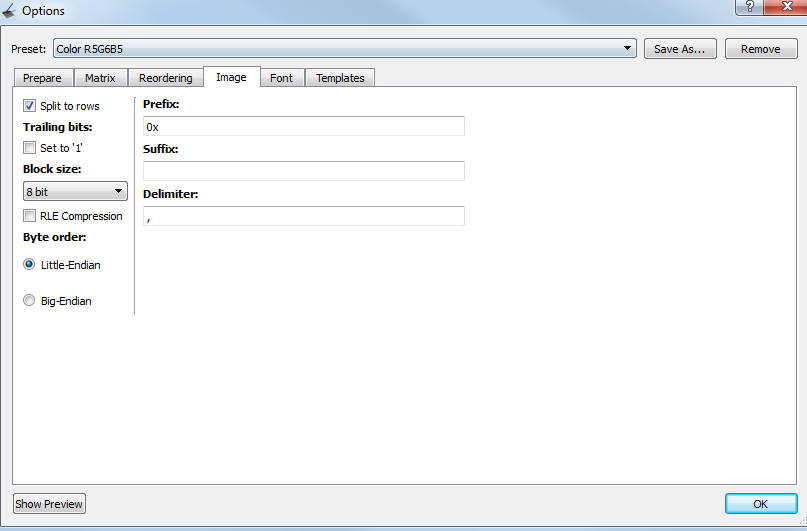
Image character array content

This text file will be transferred/copied to the uSD card for the SD shield of the Arduino.

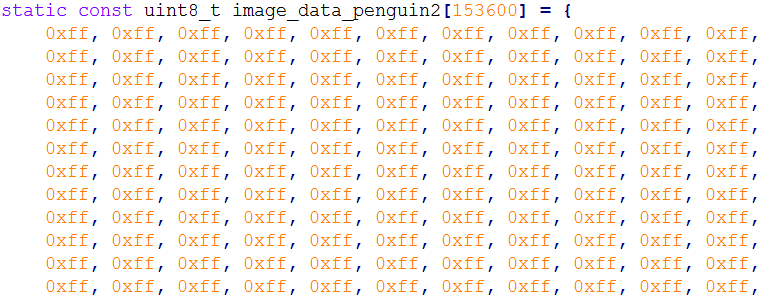
Using File->Open, you can open an existing image:



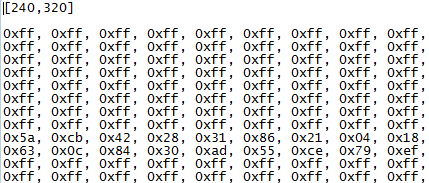
These are the conversion settings under Options-> Conversion:



If the setup is ok then click File-> Convert or Ctrl+P. The converted output is can be viewed using notepad++.



Only the contents of the array are to be copied to the text file and the content of the text file should look like the image below:



The first line is the width and height of the image and next line contains the character array.

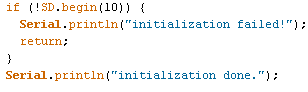
# Program the Arduino Host

A thorough understanding of the application note [**Serial Connection to an Arduino Host**](http://www.4dsystems.com.au/appnote/4D-AN-00092/) is required before attempting to proceed further beyond this point. [**Serial Connection to an Arduino Host**](http://www.4dsystems.com.au/appnote/4D-AN-00092/) provides all the basic information that a user needs to be able to get started with Serial Environment and Arduino. The following is a list of the topics discussed in [**Serial Connection to an Arduino Host**](http://www.4dsystems.com.au/appnote/4D-AN-00092/).

* Understanding the Arduino Demo Sketch (without Message Logging)
* How to download and install the Serial-Arduino library
* How to open a serial port for communicating with the display and how to set the baud rate
* The Acknowledgement Byte
* The Error Handling Routine
* How to set the Timeout Limit
* How to set the baud Rate
* How to reset the Arduino Host and the Display
* How to let the Display Start Up
* How to set the Screen Orientation
* How to Clear the Screen
* The uSD Card Mount Routine
* Understanding the Arduino Demo Sketch (with Message Logging)

Discussion of any of these topics is avoided in other ViSi-Genie-Arduino application notes unless necessary. Users are encouraged to read [**Serial Connection to an Arduino Host**](http://www.4dsystems.com.au/appnote/4D-AN-00092/)first.

## Initialize SD shield

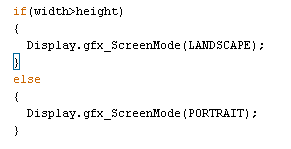


This code initializes the SD library and card. The function begin(cspin) of the SD library requires one parameter. The cspin parameter is the pin connected to the chip select line of the SD card; defaults to the hardware SS line of the SPI bus.

## Get Width and Height



This code parses the width and height “[width, height]” in the text file and stores it in a variable.

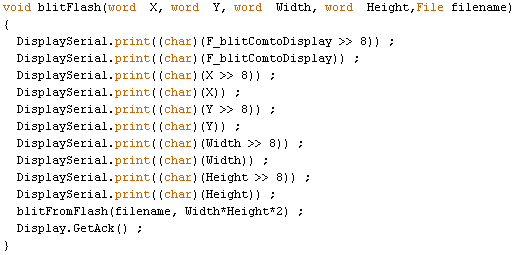


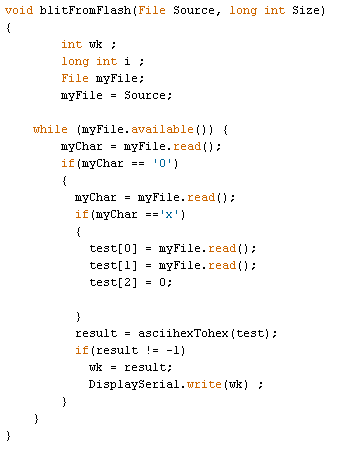
After the width and height is determined. Using those values, the Screen Mode or the Orientation of the image can be determined. If the width is greater than the height then the image is landscape. If the height is greater than the width then the image is portrait.

## Perform BLIT



This calls on the created function blitFlash to perform block image transfer. The first two parameters are the starting point. The width and height parameters are the values parsed from the text file. The myFile variable is the filename of the text file that contains the character array of the image.





The Blit Com to Display command will BLIT (Block Image Transfer) 16 bit pixel data from the Com port on to the screen. This requires five parameters. The x,y parameters specifies the horizontal and vertical position of the top-left corner of the image to be displayed. The width parameter is the width of the image to be displayed. The height parameter is the height of the image to be displayed. The data parameter is the 16 bit pixel data to be plotted on the Display screen. The data parameter must be complete for it to be executed. This function was edited because arduino cannot store a large data that contains the 16 but pixel data.

The blitFlash and blitFromFlash functions are modified functions of BlitComtoDisplay. The modification is made so that the data parameter will come from the uSD card serially accessing by arduino and then serially transfers to the display.

The blitFlash sends commands to the serial for the display to perform its “BlitComtoDisplay” function. After the commands are sent, the function blitFromFlash function is called for the host to start sending serially the character array in the uSD card.

The blitFromFlash function accesses the text file in the SD card. This function reads the character array and then writes it serially to the display.

# Connect the 4D Display Module to the Arduino Host

Refer to the section “**Connect the Display Module to the Arduino Host”** of the application note [**Serial Connection to an Arduino Host**](http://www.4dsystems.com.au/appnote/4D-AN-00092/)for the following topics:

* Using the New 4D Arduino Adaptor Shield (Rev 2.00)
  + Definition of Jumpers and Headers
  + Default Jumper Settings
  + Change the Arduino Host Serial Port
  + Power the Arduino Host and the Display Separately
* Using the Old 4D Arduino Adaptor Shield (Rev 1)
* Connection Using Jumper Wires

# Proprietary Information

The information contained in this document is the property of 4D Systems Pty. Ltd. and may be the subject of patents pending or granted, and must not be copied or disclosed without prior written permission.

4D Systems endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission. The development of 4D Systems products and services is continuous and published information may not be up to date. It is important to check the current position with 4D Systems.

All trademarks belong to their respective owners and are recognised and acknowledged.

# Disclaimer of Warranties & Limitation of Liability

4D Systems makes no warranty, either expresses or implied with respect to any product, and specifically disclaims all other warranties, including, without limitation, warranties for merchantability, non-infringement and fitness for any particular purpose.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

In no event shall 4D Systems be liable to the buyer or to any third party for any indirect, incidental, special, consequential, punitive or exemplary damages (including without limitation lost profits, lost savings, or loss of business opportunity) arising out of or relating to any product or service provided or to be provided by 4D Systems, or the use or inability to use the same, even if 4D Systems has been advised of the possibility of such damages.

4D Systems products are not fault tolerant nor designed, manufactured or intended for use or resale as on line control equipment in hazardous environments requiring fail – safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly to death, personal injury or severe physical or environmental damage (‘High Risk Activities’). 4D Systems and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

Use of 4D Systems’ products and devices in 'High Risk Activities' and in any other application is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless 4D Systems from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any 4D Systems intellectual property rights.