Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

EPSON
EXCEED YOUR VISION

S5U13781R01C100 Shield Graphics Library Users Guide

Document Number: X94A-B-001-01

SEIKO EPSON CORPORATION

Rev. 1.0

NOTICE

Table of Contents

1 Introduction

This document introduces the user to the S5U13781R01C100 Shield Graphics Library. The S5U13781R01C100 Shield Graphics Library is a software library designed to simplify the process of displaying graphics and text to a panel connected to a S5U13781R01C100 Shield.

The S5U13781R01C100 Shield is designed to be used with the Arduino Due microcontroller board. For details on the Arduino Due, refer to the Arduino website at www.arduino.cc/. For further details on the S1D13781, or the S5U13781R01C100 Shield, visit the Epson Electronics America Website at vdc.epson.com.

Note:
The S5U13781R01C100 Shield TFT Board can also be used to evaluate the low cost S1D13L01 LCD Controller which provides a similar feature set as the S1D13781. The S1D13L01 is designed as a lower cost option with a streamlined feature set. The following table shows the main differences between the S1D13781 and S1D13L01. For a complete feature list for each LCD controller, refer to the S1D13781 Hardware Functional Specification and S1D13L01 Hardware Functional Specification available at vdc.epson.com.

Feature Differences	S1D13L01	S1D13781
Display Interfaces	Active TFT Displays only	Passive STN and Active TFT Displays
BitBLT Support	SW BitBLT	Hardware BitBLT Engine with: - Move BitBLT - Solid Fill BitBLT
Temperature Range	-40 to 85° C	-40 to 85° C or -40 to 105° C
Package	QFP15 128-pin	QFP15 100-pin

We appreciate your comments on our documentation, Please contact us via email at documentation@eea.epson.com.

2 Requirements

The S5U13781R01C100 Shield Graphics Library is designed to simplify the process of displaying graphics and text to a panel connected to a S5U13781R01C100 Shield TFT board that is connected to a Arduino Due Controller board.

Before installing the S5U13781R01C100 Shield Graphics Library, ensure that you have the following available:

- S5U13781R01C100 Shield TFT board
- Arduino Due Controller board
- LCD panel (default configuration is 480x272 @ 24bpp)
- Arduino Sketch IDE software v1.6.2 or greater (see Note)
- S5U13781R01C100 Graphics Library package
- Micro USB cable (to power the Due and optionally use the Serial Monitor)

Note:
The Arduino Sketch IDE software requires a platform running Windows, Mac OS X, or Linux. For specific requirements and installation instructions, refer to the Arduino website at www.arduino.cc/en/Main/Software.

Please check the Epson Electronics America Website at vdc.epson.com for the latest revision of the Graphics Library before beginning any development.

3 Installation

Installing the S5U13781R01C100 Shield Graphics Library requires two steps:

- connecting the hardware components
- installing the software

3.1 Hardware Installation

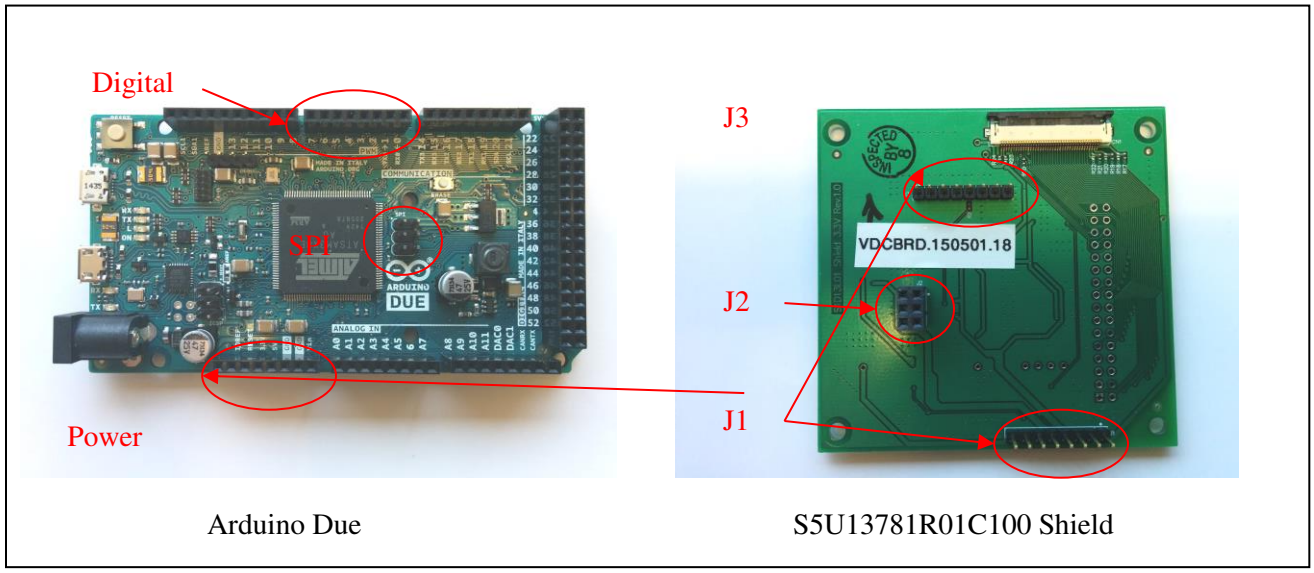
Before installing the S5U13781R01C100 Shield Graphics Library software package, connect your hardware components as shown in the following instructions.

3.1.1 Connecting S5U13781R01C100 Shield to Arduino Due

To properly connect the S5U13781R01C100 Shield to the Arduino Due, the following headers from each board must be connected together.

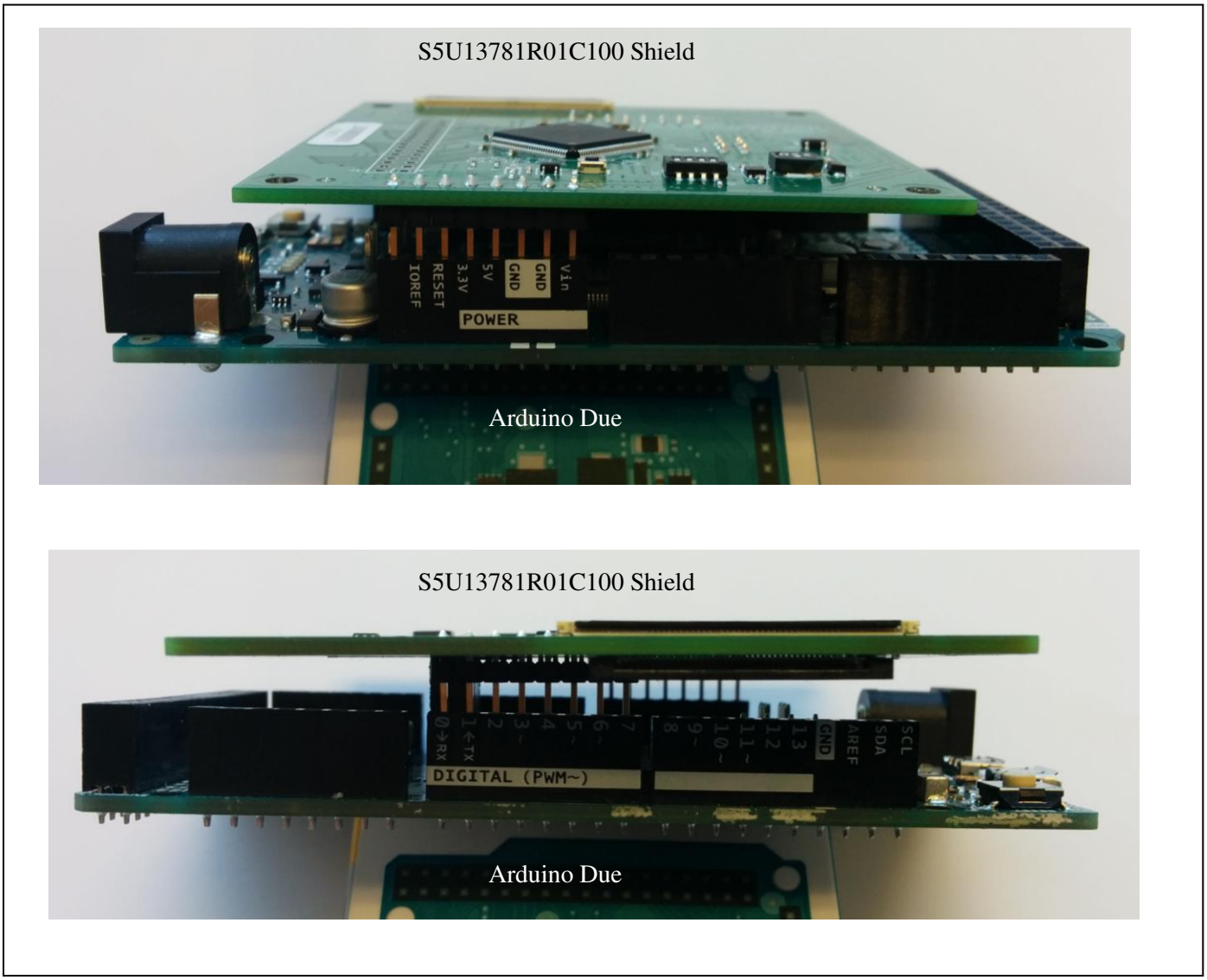
S5U13781R01C100 Shield		**Arduino Due**
J1 Header	←→	"Power" Socketr
J2 Header	←→	SPI Socket
J3 Header	←→	"Digital" (PWML) Socket

The headers and sockets for each board are identified in the following image.



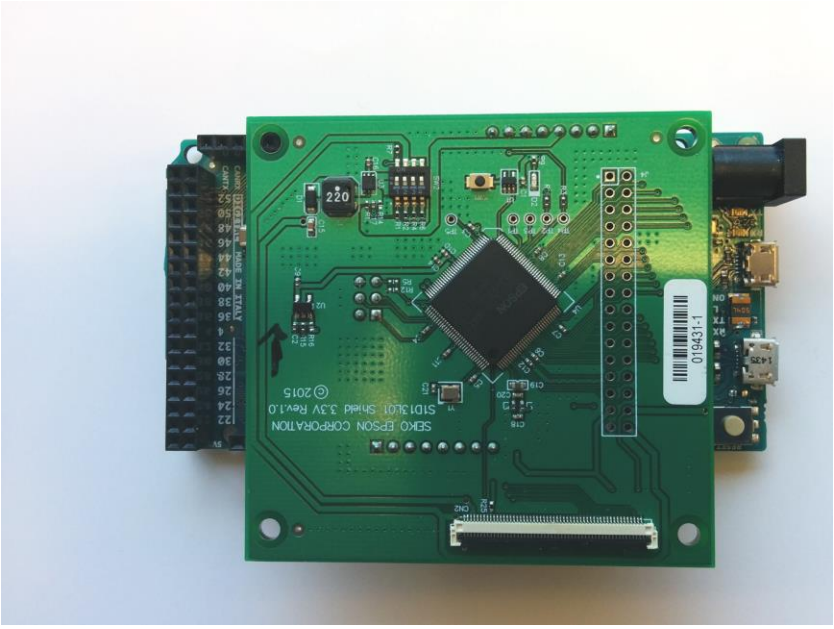
Hardware Connector Locations

To connect the boards, align the boards as shown in the following image, and then gently press the boards together. If the boards are aligned correctly, they should slide together smoothly until the header pins are completely in the corresponding socket.



Mounting the S5U13781R01C100 Shield and Arduino Due

Once the boards are connected, they should look similar to the image below.



S5U13781R01C100 Shield and Arduino Due Connected

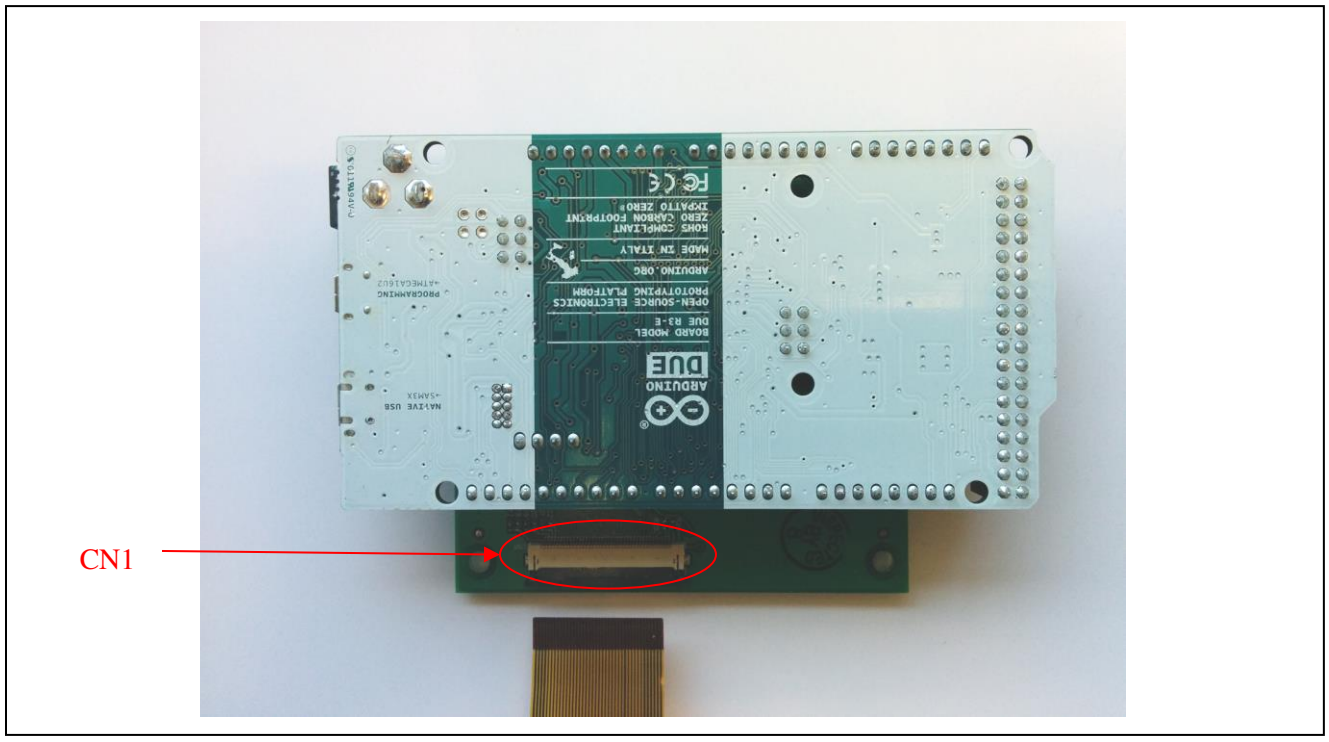
3.1.2 Connecting the LCD Panel

Once the two boards are connected, the LCD Panel can be connected to the S5U13781R01C100 Shield board. The default configuration of the S5U13781R01C100 Shield Graphics Library provides support for a Newhaven Display 480x272 LCD panel (NHD-4.3-480272EF-ATXL#).

The connector for the LCD panel is located on the side of the S5U13781R01C100 Shield board that connects to the Arduino Due. The connector is FPC connector CN1.

To connect the LCD panel:

1. Open connector CN1 by gently pulling the dark colored tabs towards the edge of the board.
2. Slide the flat panel cable into connector CN1.
3. Close connector CN1 by pushing the dark colored tabs back into place.



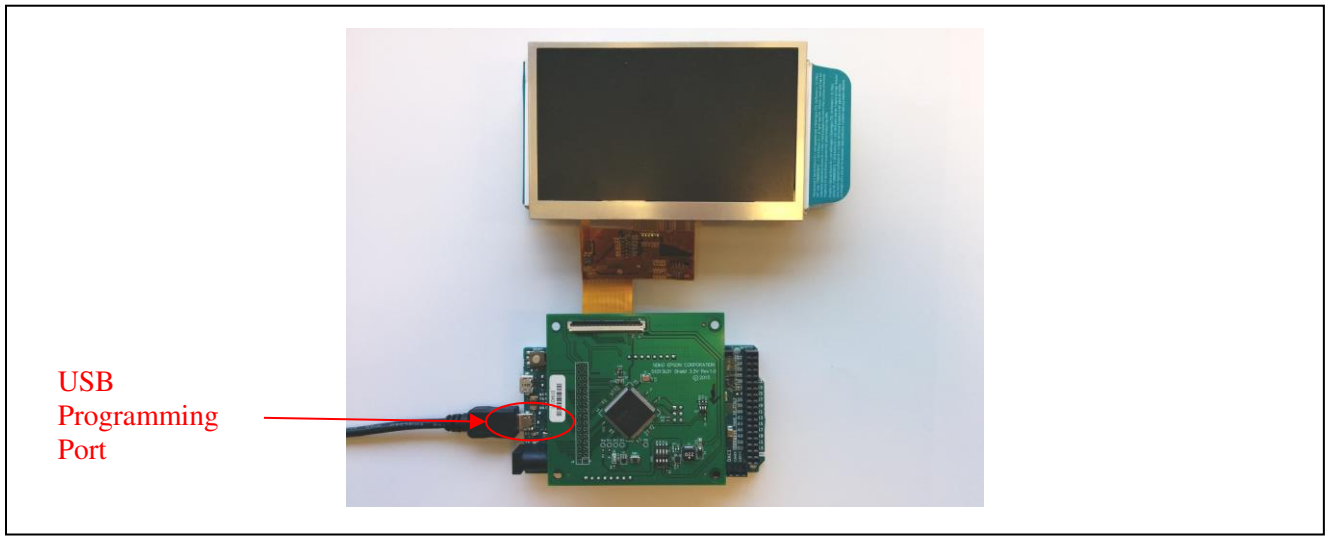
S5U13781R01C100 Shield LCD Connector

Once the panel cable is secure, carefully turn the board and panel over so that the panel is facing up.

3.1.3 Connecting the Arduino Due to Development Platform

Once all the hardware components are connected, plug a Micro USB cable between your development platform and the Programming Port on the Arduino Due.

This will provide power for the Arduino Due and optionally provide a serial monitor feature for debugging purposes.



Finished Hardware Installation

3.2 Software Installation

In order to prepare the development platform for use with the S5U13781R01C100 Shield Graphics Library, the following steps are required.

1. Install Arduino Sketch IDE
2. Install the Arduino SAM boards (includes the Due)
3. Install S5U13781R01C100 Graphics Library and example sketches
4. Use the Sketch IDE to compile example sketches and upload to the Arduino Due

3.2.1 Installing Arduino Sketch IDE

The S5U13781R01C100 Shield Graphics Library is designed to work with the Arduino Sketch IDE. It was developed and tested using Sketch v1.6.2.

Sketch requires a platform running Windows, Mac OS X, or Linux. If you do not have Sketch installed on your development platform, please visit the Arduino website and download the version of Sketch compatible with your operating system. Once Sketch is installed on your development platform, proceed to the next step.

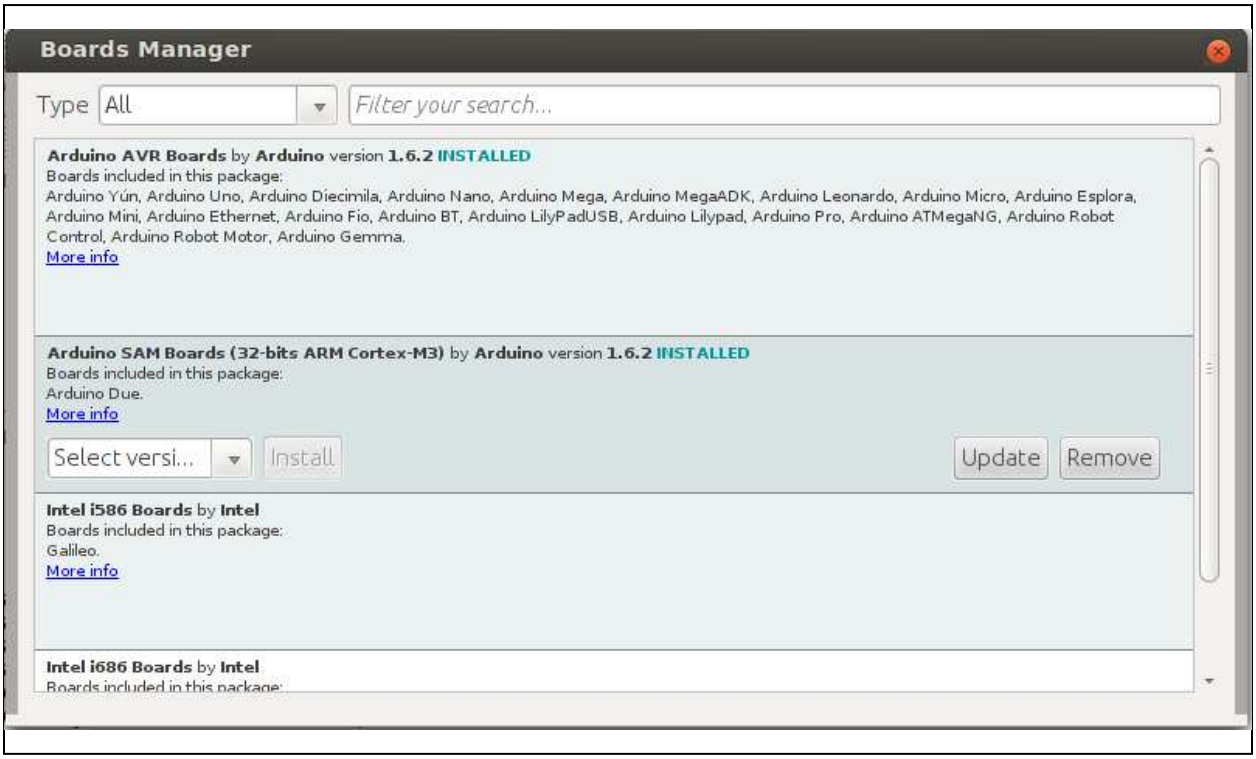
For specific requirements and installation instructions, refer to the Arduino website at www.arduino.cc/en/Main/Software.

3.2.2 Installing the Arduino SAM Boards

The default installation of Arduino Sketch may not include support for the Arduino Due. To confirm whether Due support is installed click on "Tools->Board->Boards Manager..." on the Sketch menu.

Arduino Sketch: Loading Boards Manager

This will display the Boards Manager window as shown below.



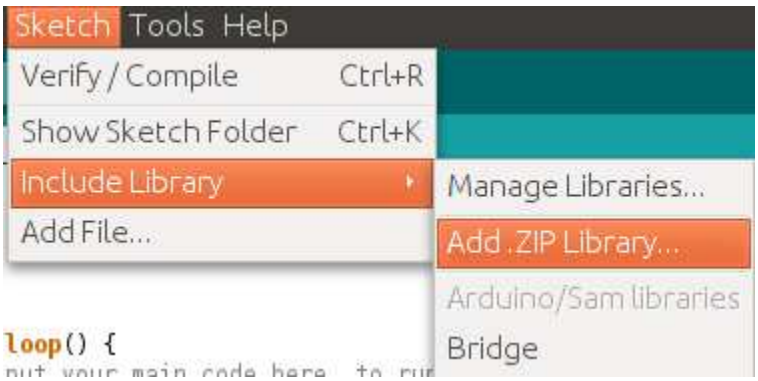
Arduino Sketch: Installing SAM Boards Package

Confirm whether the “Arduino SAM Boards (32-bits ARM Cortex-M3)” board package is installed. If the package is not installed, install the version that matches your version of the Sketch IDE. Once the SAM Boards package is installed, proceed to the next step.

3.2.3 Installing the S5U13781R01C100 Shield Graphics Library

The S5U13781R01C100 Shield Graphics Library is available as a .zip archive that includes the Graphics Library in another .zip file and a folder of example sketches. Unzip the files to a temporary location on your development platform.

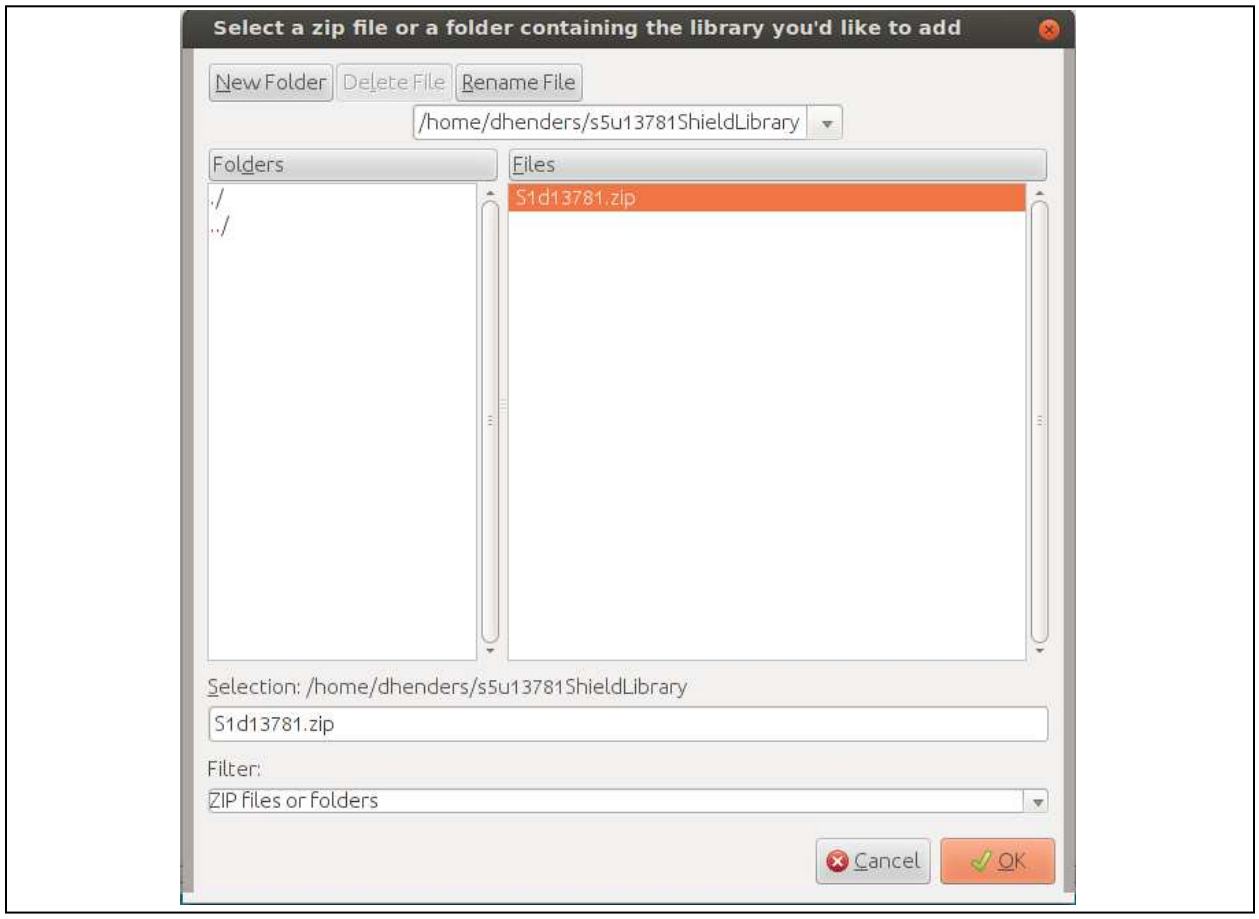
The Sketch IDE can directly import the S5U13781R01C100 Shield Graphics Library, so click on "Sketch->Include Library->Add .ZIP Library..." on the Sketch menu.



Arduino Sketch: Add .ZIP Library

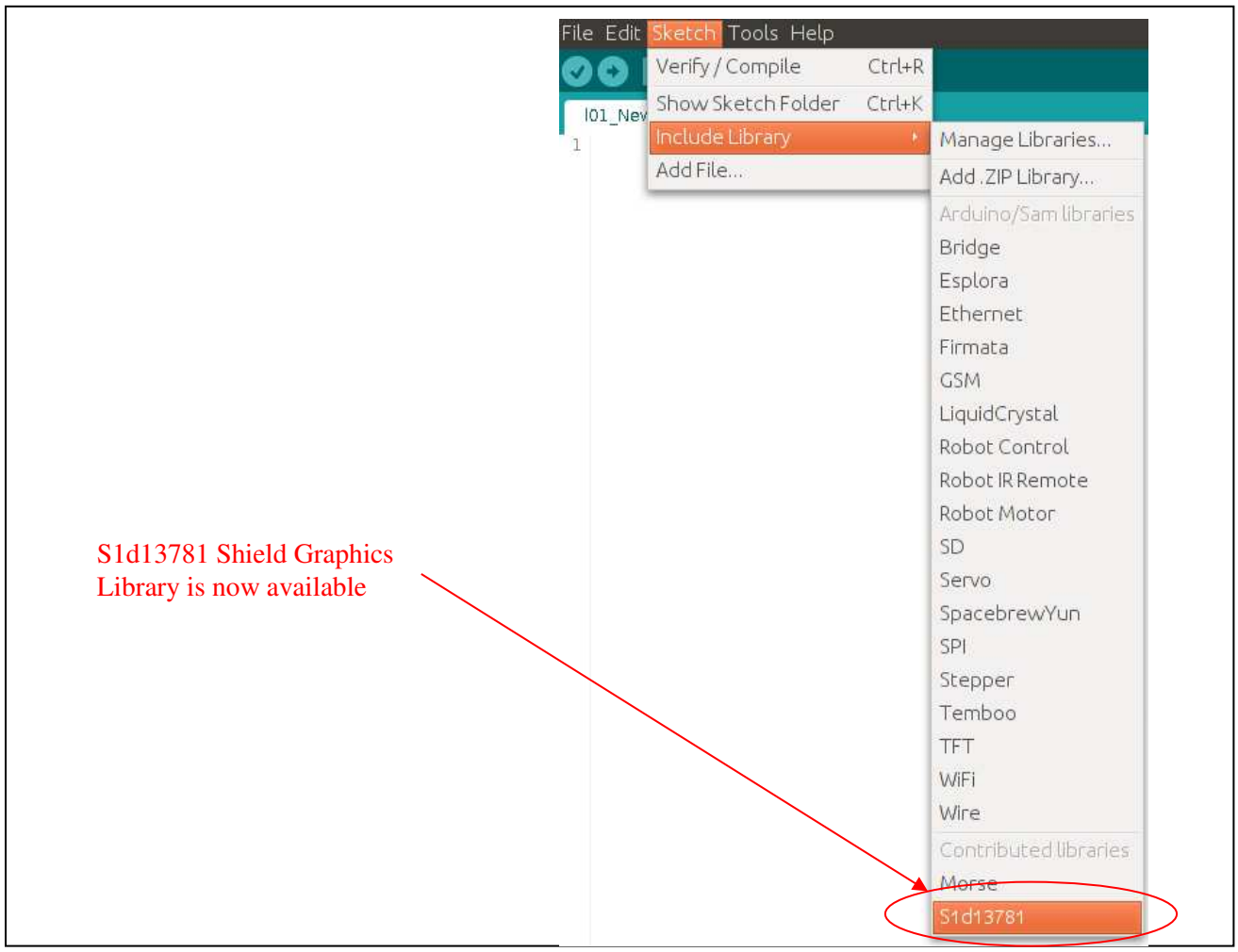
This will display the section window. Navigate to the location where the “S1d13781.zip” file was unzipped, select it, and click OK. This will install the Graphics Library into the Sketch IDE.

Note
If re-installing a modified or updated version of an existing library, the folder containing the existing library files must be removed from the “Arduino/libraries” folder before re-installing the new library.



Arduino Sketch: Loading S5U13781R01C100 Shield Graphics Library

To confirm that the S5U13781R01C100 Shield Graphics Library is installed, click on “Sketch->Include Library” and look for the “S1d13781” entry at the bottom of the list of available libraries.

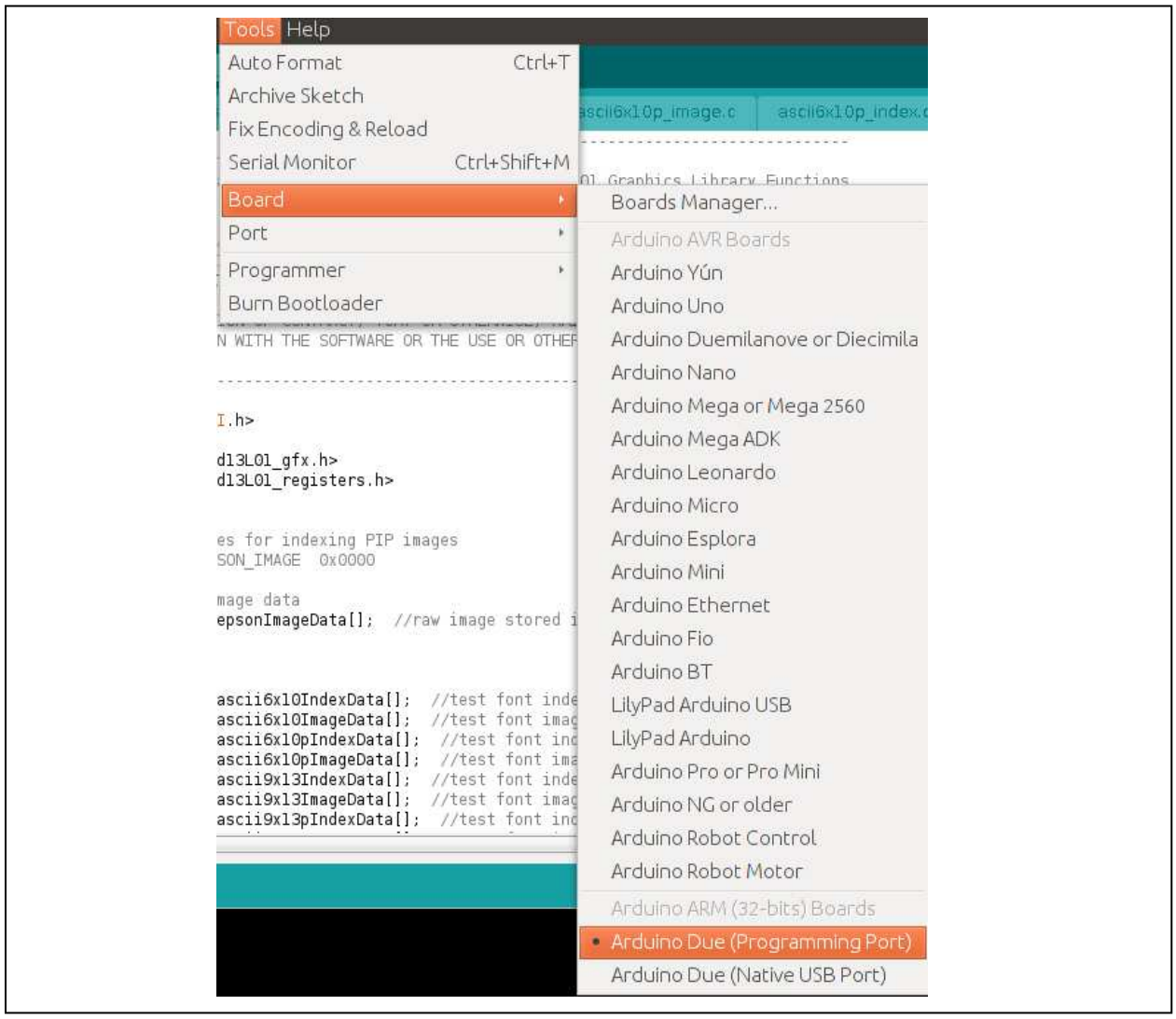


Arduino Sketch: Confirm S1d13781 Shield Graphics Library

3.2.4 Compiling and Running Example Sketch

Before we run one of the example sketches, we need to set the Board and Port settings in the Sketch IDE. The Board and Port settings tell the Sketch IDE which Arduino product is being used and how to communicate with it.

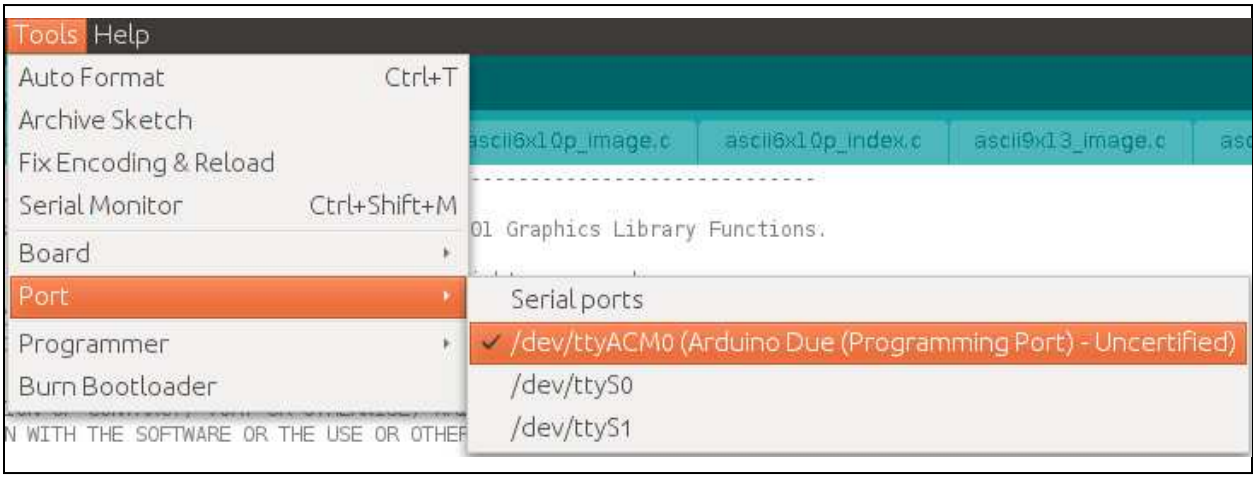
To set the Board for the Arduino Due, click "Tools->Board->Arduino Due (Programming Port)" as shown in the following image.



Arduino Sketch: Setting the Board

To set the Port for the Arduino Due, click “Tools->Port->Arduino Due (Programming Port)” as shown in the following image.

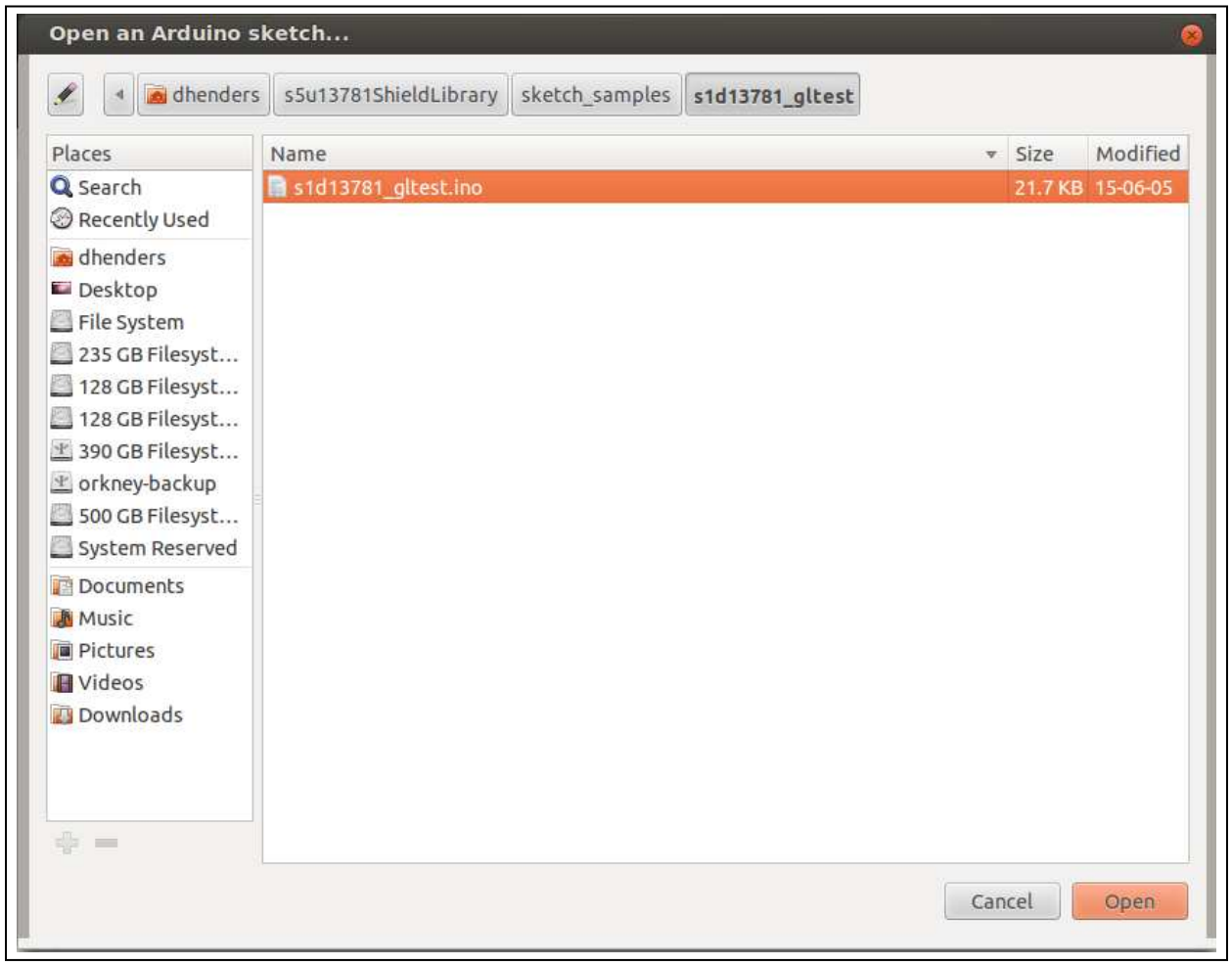
Note:
The port that the Arduino Due is located on may differ according to the operating system of the development system. For issues regarding USB port connections, please refer to the Arduino website at www.arduino.cc/.



Arduino Sketch: Setting the Port

Now that the S5U13781R01C100 Shield Graphics Library is installed, we can run one of the example sketches. To open the “781_gltest” example sketch, click on “File->Open…”, navigate to the folder where the example sketches were unzipped, select the “781_gltest.ino” sketch file, and click Open. This will open an example sketch that demonstrates some of the capabilities of the S1D13781 LCD controller and the S5U13781R01C100 Shield Graphics Library.

Note:
The default configuration assumes that a Newhaven Display 480x272 LCD panel is connected to the S5U13781R01C100 Shield.



Arduino Sketch: Loading an Example Sketch

Once the example sketch is loaded, it can be compiled and uploaded to the Arduino Due. If you simply want to check modifications or new code for errors, click the “Verify” button on the Sketch Toolbar. If you want to upload the application to the Arduino Due and run it, click the “Upload” button. Upload will verify the application code and then upload the required binary images to the Arduino Due.

Note:
The Upload stage will fail if the Sketch “Board” and “Port” settings are not configured correctly. For these settings, see the instructions earlier in this section.

The following image identifies the buttons that on the Sketch toolbar.

Arduino Sketch: Important Toolbar Functions

For both the “Verify” and “Upload” options, results and errors messages are shown in the message window of the Sketch IDE. If the upload to the Arduino Due completes successfully, the 781_gltest demo should start and provide examples of the drawing functions available in the S5U13781R01C100 Shield Graphics Library.

4 Using the S5U13781R01C100 Shield Graphics Library with Sketch

4.1.1 Modifying an Existing Sketch

There are several example sketches included with the S5U13781R01C100 Shield Graphics Library which attempt to demonstrate different concepts. For example:

- 781_gltest.ino – demonstrates some of the graphics functions available in the Graphics Library
- 781_RegisterAccessExample – demonstrates how to read/write S1D13781 registers directly
- 781_MemoryAccessExample – demonstrates how to read/write S1D13781 memory directly

In order to modify a sketch and try out a new idea, simply add new line in the Sketch loop() function that calls one of the Graphics Library functions. For example, if we want to display a green line from position 20,20 to position 100,100 on the LCD panel display, we could add the following line(s).

```
result = lcdc.drawLine( S1d13781_gfx::window_Main,20,20,100,100,0x0000FF00 );
sleep(2000);
```

For the above example:

- result is the return value from the drawLine() function. Many Graphics Library functions return a value with either the requested value, or a value that can be tested for errors.
- lcdc is the instance of the S1d13781_gfx class that is used in the example sketches.
- drawLine is the method to be called.
- the parameters passed to drawLine determine the destination window, line end positions, and color of the line.
- sleep(2000) is added so that we will have a chance to see the effect of our function call before anything else happens

For details on the methods available as part of the S5U13781R01C100 Shield Graphics Library, refer to the Library Reference section.

4.1.2 Creating a New Sketch

When creating a New sketch certain elements must be added to the basic template. The following code shows the additions which are explained below.

```
/*-----------------------------------------------------------------
 * new_sketch.ino
 * Example of a new sketch using the S5U13781R01C100 Graphics
 * Library Functions.
 *-----------------------------------------------------------------*/

#include <SPI.h>

#include <S1d13781_gfx.h>
#include <S1d13781_registers.h>


//create an instance of S1d13781 for us to work with
S1d13781_gfx lcdc;

void setup() {
  //start serial for serial monitor
```

```
  Serial.begin(9600);


  //start the S1d13781 library
  lcdc.begin();

  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

A new sketch requires the following additions to work with the S5U13781R01C100 Shield Graphics Library.

- #include <SPI.h> - The SPI header is required because the S5U13781R01C100 Shield relies on SPI to interface with the Arduino Due.
- #include <S1d13781_gfx.h> - This header includes the S5U13781R01C100 Shield library classes.
- #include <S1d13781_registers.h> - This header includes the #defines for the S1D13781 registers. When accessing S1D13781 registers it is suggested that the defined constants are used to avoid unintended access attempts to un-defined registers.
- "S1d13781_gfx lcdc;" – This line creates an instance of the S1d13781_gfx class. lcdc is what is used for the example sketches, but this name can be anything that is appropriate for your application.
- "Serial.begin(9600);" – This line is optional, but provides useful debugging capabilities which allow messages to be sent from your application to the Serial Monitor (see Using the Serial Monitor).
- "lcdc.begin();" – This line starts the S5U13781R01C100 library and performs the following functions:
    1. Setup the SPI interface using by the S5U13781R01C100 Shield.
    2. Initializes the registers according to the settings in S1d13781_init.h.
    3. Leaves the S1D13781 in a powered on state (ready to go).

4.1.3 Using the Serial Monitor

When the Arduino Due is connected through the Programming Port, the Serial Monitor function is available. The Serial Monitor can be started using the "Serial Monitor" button on the Sketch toolbar, or by clicking "Tools->Serial Monitor" on the Sketch menu.

The Serial Monitor allows information and messages from the application running on the Arduino Due to be sent to and displayed by the Sketch IDE. This can be a helpful tool when debugging modified application code or creating new applications. The example sketches included in the S5U13781R01C100 Shield Graphics Library include examples of how to use the Serial class to output information to the Serial Monitor.

For more information on using the Serial Class and Serial Monitor, refer to the Arduino language reference at www.arduino.cc/en/Reference/.

4.1.4 Using Fonts with the S5U13781R01C100 Shield Graphics Library

The S5U13781R01C100 Shield Graphics Library supports text drawing using a programmable font. The font relies on two components:

- A 1 bpp image file stored as a .pbm graphics file (binary)
- A portable font index stored as a .pfi file (binary)

The Graphics Library package provides some sample fonts and includes both the .pbm image file and .pfi index file as binary files.

Note:
The .pfi files are also stored in a sub-folder as text files for reference. For more information on the font index files and information useful for creating custom fonts, refer to the Readme.txt in the sample font folder.

All the sample fonts included in the package are created by Epson and are free to modify and/or use.

- Ascii4x6
- Ascii4x6p (proportional font)
- Ascii6x10
- Ascii6x10p (proportional font)
- Ascii7x11
- Ascii7x11p (proportional font)
- Ascii9x13
- Ascii9x13p (proportional font)
- AsciiCaps4x6
- AsciiCaps4x6p (proportional font)
- Latin6x10
- Latin6x10p (proportional font)
- LineDraw6x10 (includes line draw graphics suitable for line art buttons)

When using a font within a Sketch application, a simple method to provide the desired font to the application is to convert the binary .pbm and binary .pfi files into a simple byte arrays. If the arrays are stored as "C" source, they can be copied into the Sketch application folder and referenced from the Sketch application to be passed to the S1d13781_gfx::createFont() method when necessary.

An example is included in the sample sketch "781_gltest.ino" which includes several of the sample fonts. To make use of the external byte arrays, the following lines could be added to the Sketch application.

```
//test fonts
extern byte ascii9x13IndexData[];  //test font index data
extern byte ascii9x13ImageData[];  //test font image data
```

Then to create the font, send the data and the number of bytes for each array.

```
testfont = lcdc.createFont(ascii9x13ImageData, 1453, ascii9x13IndexData, 498);
```

Once the font is created it can be used with methods such as drawText() and drawMultiLineText(). When the font is not required anymore, call the freeFont() method to free the font resources.

For further information on the Font methods, refer to the Library Reference section.

5 Understanding the Graphics Library

The S5U13781R01C100 Shield Graphics Library is a collection of C++ methods organized into two classes: S1d13781 and S1d13781_gfx. When integrated into the Arduino Sketch IDE they provide hardware access and simple graphics routines which enable users to quickly display graphics and text to a LCD panel connected to the S5U13781R01C100 Shield.

5.1.1 Library Structure

The S5U13781R01C100 Shield Graphics Library is organized into the following files:

- S1d13781.h – The header file for the S1d13781 class. It is a good place to get an overview of hardware oriented functions of the Graphics Library. This file also includes some constants that configure the SPI interface used between the S5U13781R01C100 Shield and the Arduino Due.
- S1d13781.cpp – The source file for the S1d13781 class. It includes the source for the methods that allows access to the hardware level functions of the S1D13781 LCD controller. This includes functions such as direct register and memory access, initialization of the S1D13781, and setup of SPI used for the interface between the S5U13781R01C100 Shield and the Arduino Due.
- S1d13781_gfx.h – The header file for the S1d13781_gfx class. It provides an overview of the graphics and text methods implemented in the Graphics Library.
- S1d13781_gfx.cpp – The source file for the S1d13781_gfx class. It includes the source for the graphics and text display methods available in the Graphics Library.
- S1d13781_init.h – This header file includes a structure that contains the values used to initialize the S1D13781 hardware registers.
- S1d13781_registers.h – This header file includes #defines for the S1D13781 hardware registers. When accessing S1D13781 registers it is suggested that the defined constants are used to avoid unintended access attempts to un-defined registers.
- keywords.txt - A file required for Library support in the Arduino Sketch IDE. Any new classes and/or methods should be added to this file. For further information on keywords.txt, refer to the Library Tutorial at www.arduino.cc/.

5.1.2 Modifying the Graphics Library

Full source code is provided for the S5U13781R01C100 Shield Graphics Library allowing customization and modification by the user. The Graphics Library source is not intended to be modified from within the Sketch IDE, so an external code editing tool should be used.

Once the Graphics Library is installed in the Sketch IDE, it can be modified directly in the "Arduino/libraries/S1d13781" folder. Alternately, it can be modified "off-tree" and re-installed as a .ZIP Library. However, for the re-installation method, the existing S1d13781 library must be removed from the "Arduino/libraries" folder before re-installing the updated Graphics Library.

For full Sketch IDE support, if new Classes and/or methods are added to the Graphics Library they should be added to the keywords.txt file included in the Graphics Library package. For further information on keywords.txt, refer to the Library Tutorial at www.arduino.cc/.