# Application Note

# AN_280

# 4DLCD-FT843 SampleApp Arduino Introduction

**Version 1.0**

**Issue Date: 2013-10-30**

This document introduces how to setup the FT800 Sample Application running on an Arduino - ATMega328 system using the 4DLCD-FT843 system.

The objective of the Sample Application is to enable users to become familiar with the usage of the FT800, the design flow, and display list language used to design the desired user interface or visual effect.

**Future Technology Devices International Limited (FTDI)**
Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom
Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758
Web Site: http://ftdichip.com

## Table of Contents

# 1 Introduction

The FT800 combines display, audio and touch functionality into one single chip, powered by FTDI Chip's advanced EVE technology (Embedded Video Engine). The FT800 device interfaces with a system MCU via either an SPI or I$^2$C interface. To enable customers to more easily utilize the functionality of the FT800 in a project, a Sample Application is provided here for tutorial purposes.

The sample application has been written for an ATMega328 based platform e.g. Arduino Uno or Sparkfun RedBoard (DEV-10908).

Users can read the source code of the Sample Application first, and then run the code to observe the effects. Editing the code is also encouraged to help learn the features of the FT800.

Note that although the basic project is created for ATMega328, the code relating to the creation of the screen shots could be reused in different MCU design environments. In addition, the set-up steps for the ATMega328 would be necessary for other MCU's.

This document introduces how to set up and use the Sample Application with an FTDI 4DLCD-FT843 development system- **Figure 1** in relation to conjunction with the Sparkfun RedBoard platform. Further information regarding the FT800 programming language or pseudo-code can be found in the FT800 Programmer Guide.

For 4DLCD-FT843 development board details, please refer to www.4dsystems.com.au

To learn more about Arduino Uno and its IDE, please check www.arduino.cc

To learn more about Sparkfun RedBoard (DEV-10908). Please check www.sparkfun.com

**NOTE: Any source code is provided on an "as is" basis, and is neither guaranteed nor supported.**

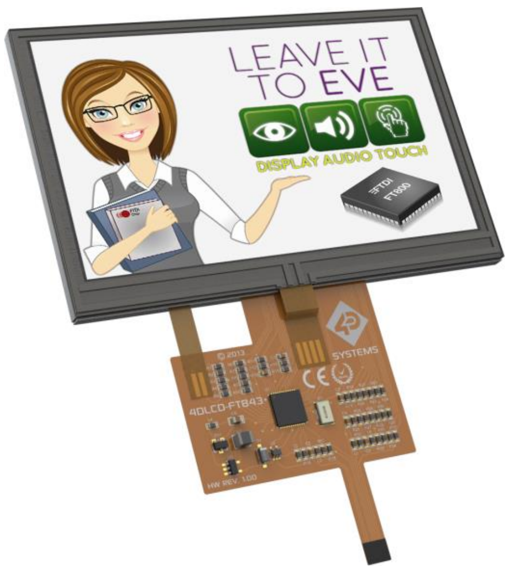


**Figure 1: 4DLCD-FT843 Display incorporating the FT800**

**Figure 2** shows the 4DLCD-FT843 Display connected to the Sparkfun RedBoard (DEV-10908). The 4DLCD-FT843 enables system designers to rapidly create high quality, human machine interfaces (HMIs). It includes a 4.3" TFT display (with 480 x 272 pixel resolution), PWM audio output (with amplifier enable), 58 synthesized sounds, a 4-wire resistive touch screen, all integrated on a flexible ribbon connector.

The 4DLCD-FT843 Display is connected to the Sparkfun RedBoard (DEV-10908) via the Arduino Display Adaptor Module (ADAM 4Display-Shield-FT843).

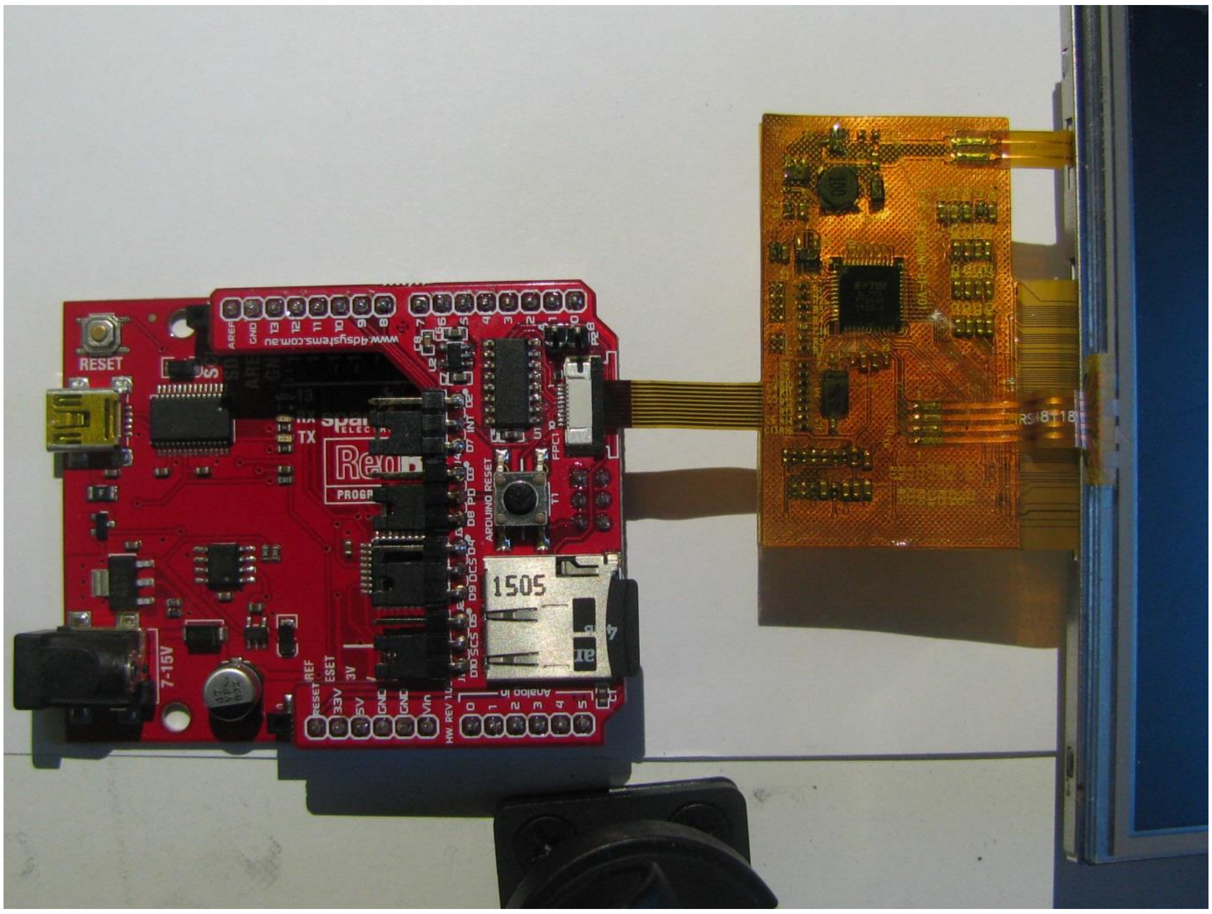


**Figure 2: 4DLCD-FT843 Display connected to the Sparkfun RedBoard**

## 1.1 Audience

This document assumes the audience has read the datasheet and Programmer Guide of the FT800. In addition, familiarity of the C/C++programming language is necessary to understand the Sample Application source code. To understand the SPI of the Arduino Platform, knowledge of ATMega328 microcontroller and IDE (Integrated Design Environment) is required.

## 1.2 Scope

The Sample Application mentioned in this document is created in Arduino IDE and runs on Sparkfun RedBoard connected to the 4DLCD-FT843 Display module. It is comprised of the source code as well as project files.

## 1.3 Overview

### 1.3.1 Hardware Block Diagram

**Figure 3** below gives the basic hardware setup with additional audio components.

The Arduino Display Adaptor Module (shown in red dashes) is used to connect the Arduino board to the 4DLCD-FT843 Display via SPI. The SampleApp code includes audio, if this is required a simple filter / amplifier circuit and speaker can added as shown in figure 3. An alternative is to connect a standalone speaker with built-in amplifier.

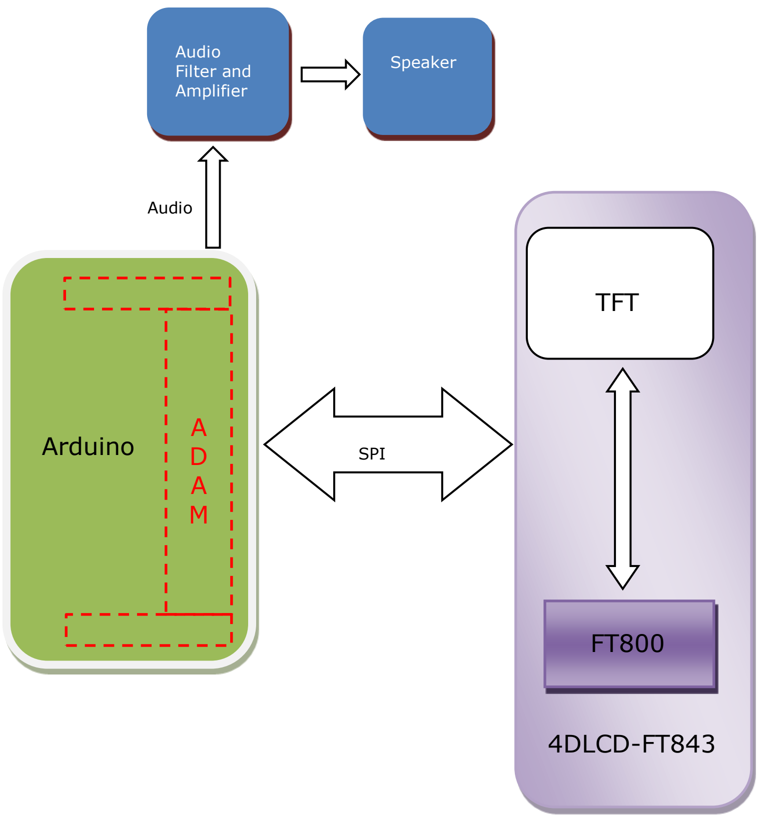


**Figure 3: Block Diagram of Hardware Setup**

### 1.3.2 Application flow

The diagram below gives the basic flow and structure for configuring the FT800 in an application.

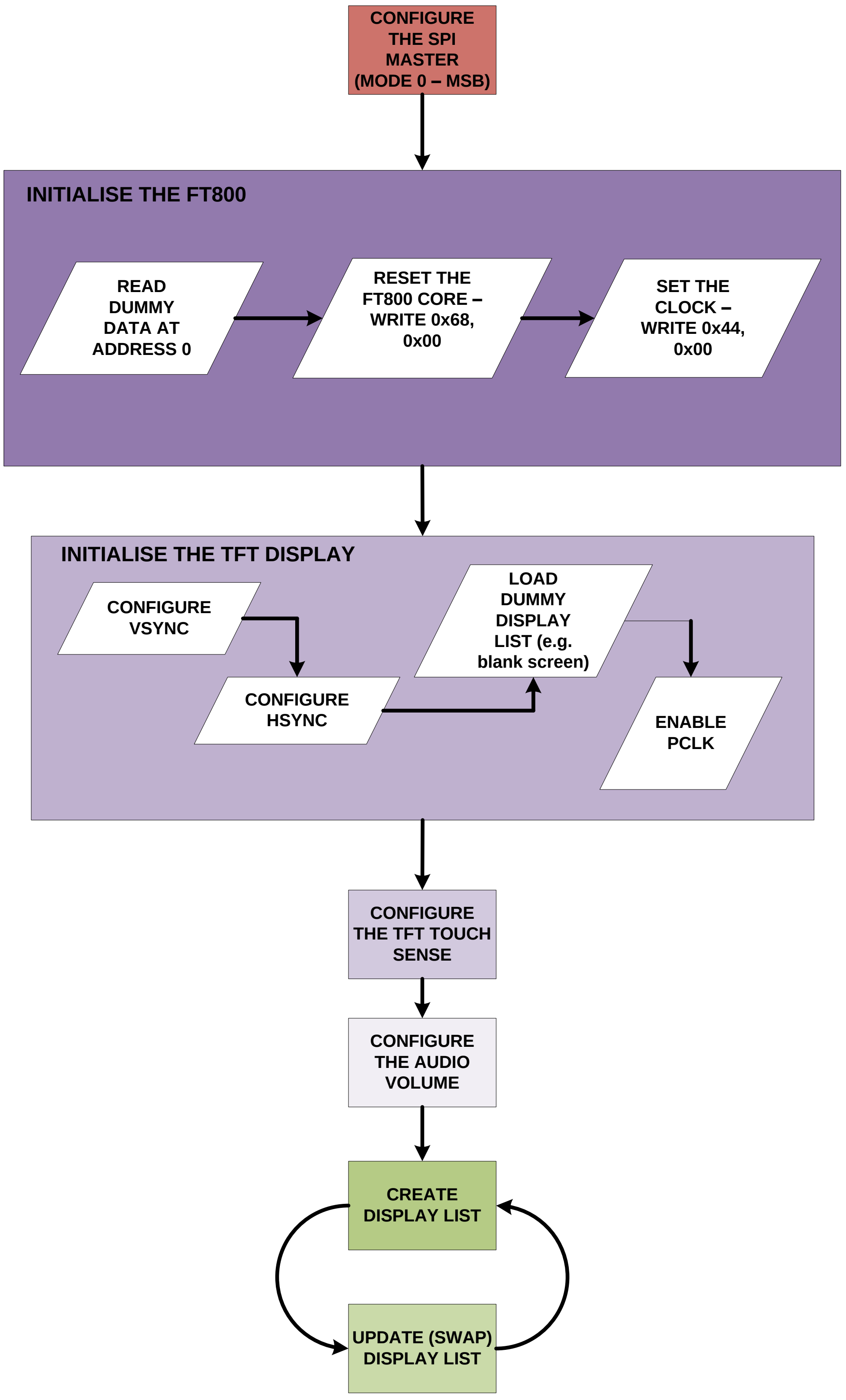

CONFIGURE THE SPI MASTER (MODE 0 – MSB)
INITIALISE THE FT800
READ DUMMY DATA AT ADDRESS 0
RESET THE FT800 CORE – WRITE 0x68, 0x00
SET THE CLOCK – WRITE 0x44, 0x00
INITIALISE THE TFT DISPLAY
CONFIGURE VSYNC
CONFIGURE HSYNC
LOAD DUMMY DISPLAY LIST (e.g. blank screen)
ENABLE PCLK
CONFIGURE THE TFT TOUCH SENSE
CONFIGURE THE AUDIO VOLUME
CREATE DISPLAY LIST
UPDATE (SWAP) DISPLAY LIST

**Figure 4: Application Flow**

### 1.3.3 Architecture

The Sample Application is designed to easily port to various platforms with SPI host functionality. Therefore, this sample application introduces one generic HAL (hardware abstraction layer) which can be used as a guideline for other platforms. Note additional processor specific HALs are being developed, check www.ftdichip.com for the latest support.

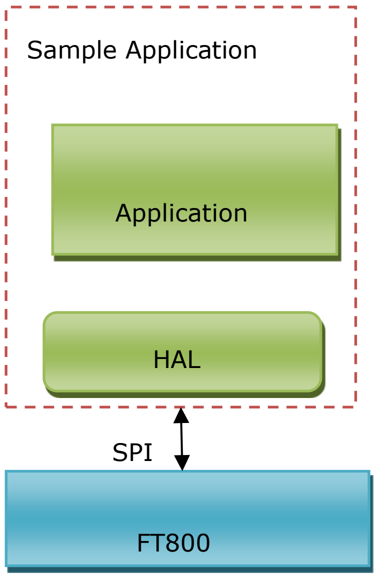


**Figure 5: Architecture diagram**

## 1.4 Hardware requirement

- 4DLCD-FT843 DISPLAY development kit and additional simple filter and amplifier circuit for Audio.
- Arduino Display Adaptor Module – ADAM (4Display-Shield-FT843)
- ATMEL MEGA 328P microcontroller - Arduino Uno or Sparkfun RedBoard (DEV-10908).
- USB cable with a Mini-B connector to program /power the Arduino.

## 1.5 Software requirement

- Arduino IDE 1.0.5
- FT800 Sample Application release package.

### 1.5.1 Software package introduction

#### *1.5.1.1 Folder introduction*

- Folder "Project\Arduino" contains all the source code and project file "SampleApp.ino".
- The other folders are not relevant to the Arduino platform.

#### *1.5.1.2 Dependency*

The Sample Application uses the SPI library provided by Arduino with Arduino IDE. Please check the Arduino website for details.

# 2 Set up steps

## 2.1 Hardware Connection

The ADAM module is mated onto the top of the Sparkfun RedBoard (DEV-10908) as shown in **Figure 6.** The 4DLCD-FT843 DISPLAY flat panel cable (FPC) connects into the FPC socket on the top of the ADAM module. USB micro cable is used for power/programming.

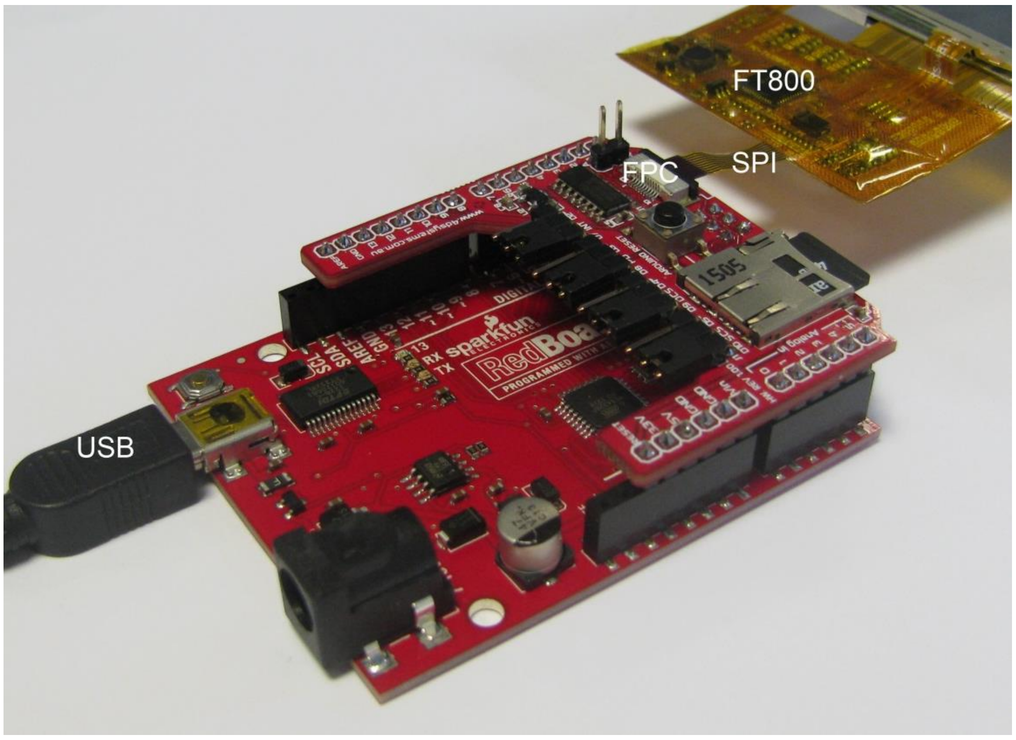


**Figure 6: Hardware connections**

### 2.1.1 ADAM - Arduino Display Adaptor Module Connections

**Figure 7** shows the Arduino Display Adaptor Module connections, **Table 1** gives a brief description of each connection. Jumpers J1 to J4 are shown in their default positions. For full details of the ADAM module (4Display-Shield-FT843) refer to the data sheet available at www.ftdichip.com

| Connector | Description |
|---|---|
| **FPC1** | 10 way Bottom Contact FPC for connecting directly to the 4DLCD-FT843 displays flex |
| **P2** | Audio output to a simple filter / amplifier circuit and speaker or a standalone speaker with built-in amplifier. |
| **RESET** | Resets the ATMega328 |
| **MICRO SD** | The micro-SD socket on the ADAM Shield enables the Arduino to access files for use with the FT800 Graphics Controller, along with being a storage media for general storage used by the Arduino. |
| **J1** | SD card Chip Select – default pin 1 and 2 |
| **J2** | Display Chip Select – default pin 1 and 2 |
| **J3** | PD – Power down display enable – default pin 1 and 2 |
| **J4** | INT - Interrupt from the Display – default pin 1 and 2 |

**Table 1**

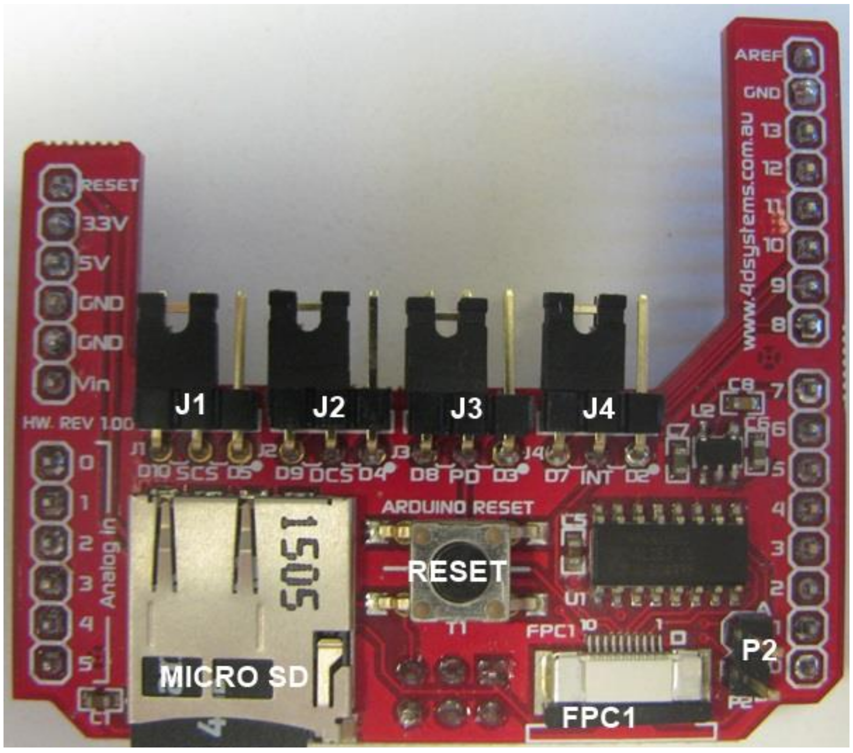
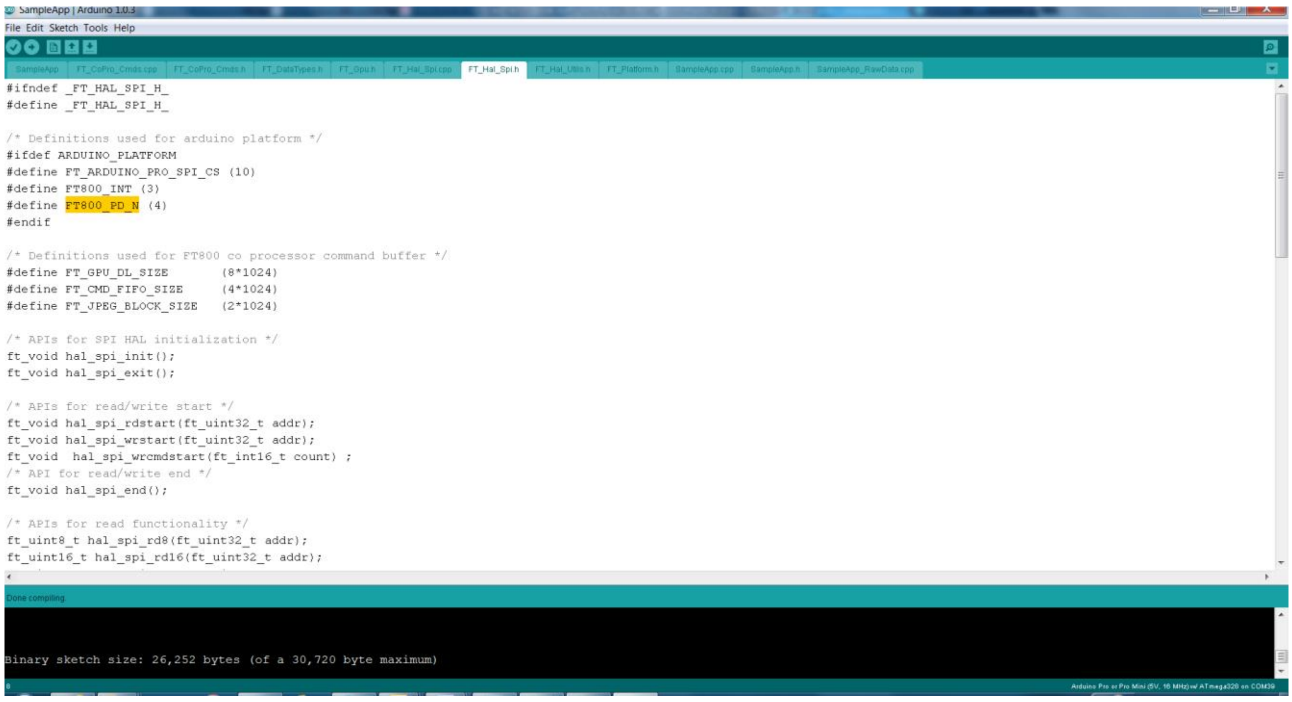


**Figure 7 ADAM module**

## 2.2 Source code build and download

Please note that all the related source code resides at folder "Project\Arduino\SampleApp".

To build the project, open the file "Project\Arduino\SampleApp\SampleApp.ino " with the Arduino IDE and the following screen will be shown:



```
#ifndef _FT_HAL_SPI_H_
#define _FT_HAL_SPI_H_

/* Definitions used for arduino platform */
#ifdef ARDUINO_PLATFORM
#define FT_ARDUINO_PRO_SPI_CS (10)
#define FT800_INT (3)
#define FT800_PD_N (4)
#endif

/* Definitions used for FT800 co processor command buffer */
#define FT_GPU_DL_SIZE        (8*1024)
#define FT_CMD_FIFO_SIZE      (4*1024)
#define FT_JPEG_BLOCK_SIZE    (2*1024)

/* APIs for SPI HAL initialization */
ft_void hal_spi_init();
ft_void hal_spi_exit();

/* APIs for read/write start */
ft_void hal_spi_rdstart(ft_uint32_t addr);
ft_void hal_spi_wrstart(ft_uint32_t addr);
ft_void  hal_spi_wrcmdstart(ft_int16_t count) ;
/* API for read/write end */
ft_void hal_spi_end();

/* APIs for read functionality */
ft_uint8_t hal_spi_rd8(ft_uint32_t addr);
ft_uint16_t hal_spi_rd16(ft_uint32_t addr);
```

```
Binary sketch size: 26,252 bytes (of a 30,720 byte maximum)
```

**Figure 8: Arduino IDE with the opened project**

Press "Ctrl+R" to rebuild and "Ctrl+U" to download the binary generated into Arduino.

For further details of downloading or programming Arduino, check the website of Arduino.

### 2.2.1 Determine the screen size

For 3.5 inch displays, ensure the compilation macro switch "#define SAMAPP_DISPLAY_QVGA" in file "Project\Arduino\FT_Platform.h"is defined.

For other development boards (4.3" and 5.0" displays), make sure the macro above is undefined within the sample application project. This will ensure the correct display resolution is selected to match the correct display size.

After correctly setting the definition, re-build the project.

### 2.2.2 Determine the group of functions to build and run

Due to the limitation of the ATMega328 flash size (32KBytes), it is impossible to put all the functions together and program them into the ATMega328. All the functions are grouped into 5 categories, which are compiled and built under following compiler switches in the file "Project\Arduino\FT_Platform.h "

```
#define SAMAPP_ENABLE_APIS_SET0
#define SAMAPP_ENABLE_APIS_SET1
#define SAMAPP_ENABLE_APIS_SET2
#define SAMAPP_ENABLE_APIS_SET3
#define SAMAPP_ENABLE_APIS_SET4
```

Users must enable one API set exclusively, by defining one of the above macro's at a time, otherwise the binary build will not fit into the ATMega328 flash.

Note the categories mentioned here are not same as the groups mentioned in section 2.2.5. The groups here are defined for the purpose of running on the ATMega328.

### 2.2.3 Source file brief

*"SampleApp.cpp"* is the main source file for the Sample Application. The main entry function is inside. It defines all the sample functions.

The functions in "SampleApp.cpp" are mostly in the form of "SAMAPP_GPU_xxx" and "SAMAPP_CoPro_xxx".

*"FT_Gpu_Hal.cpp"* defines the transportation layer functions, which provides one SPI abstraction layer to access the FT800. Editing this file allows for porting the application to alternative MCU's and compilers with minimal effort. It is more specific to the SPI master interface.

*"FT_CoPro_Cmds.cpp"* defines the APIs of the FT800 coprocessor engine commands. This file is structured to be generic and could be ported to other projects for other target MCU's.

*"FT_GPU.h"* defines the FT800 specific instruction parameters, register names and memory maps. The contents of this file relate directly to the FT800 Programmers Guide and is structured to be generic such that it could be ported to other projects for other target MCU's.

*"SampleApp_RawData.cpp"* defines the bitmap data used in sample application.

### 2.2.4 Project file brief

"SampleApp.ino" is the project file used by the Arduino IDE and it includes all the necessary files in this project.

The major functions in the sample application can be classified into the following groups according to functionality and design purpose.

### 2.2.5 Major function groups in sample application

The major functions in sample application can be classified into following group according to its functionality and design purpose.

#### *2.2.5.1 Primitives group*

The functions in this group are designed to demonstrate the usage of FT800 primitives.

An FT800 primitive is the basic drawing command e.g. Points are used to draw circles, while Lines is used for straight lines. More information on the primitives may be found in the FT800 Programmers Guide.

All the function are in the form of "SAMAPP_GPU_xxx". Here is the list:

```
      /*draw circles*/
o     SAMAPP_GPU_Points();
      /*draw a triangle*/
o     SAMAPP_Gpu_Polygon();
      /*draw lines*/
o     SAMAPP_GPU_Lines();
      /*draw rectangles*/
o     SAMAPP_GPU_Rectangles();
      /*draw bitmaps*/
o     SAMAPP_GPU_Bitmap();
      /*draws chars with different fonts*/
o     SAMAPP_GPU_Fonts();
o     SAMAPP_GPU_Text8x8();
o     SAMAPP_GPU_TextVGA();
      /*draws a bargraph*/
o     SAMAPP_GPU_Bargraph();
o     SAMAPP_GPU_LineStrips();
o     SAMAPP_GPU_EdgeStrips();
      /*example of cutting away an active area on the display*/
o     SAMAPP_GPU_Scissor();
      /*Font and Points Primitives combination*/
o     SAMAPP_GPU_FtdiString();
      /*Call and Return Primitives combination*/
o     SAMAPP_GPU_StreetMap();
      /*Additive blending of fonts*/
o     SAMAPP_GPU_AdditiveBlendText();
      /*Usage of Macro*/
o     SAMAPP_GPU_MacroUsage();
      /*Additive blending of points*/
o     SAMAPP_GPU_AdditiveBlendPoints();
```

#### *2.2.5.2 Widgets group*

The functions in this group are designed to demonstrate the FT800 graphic engine widgets, which are visual components to reduce the effort of GUI programmers.

A widget will create a complex object with one command as opposed to many e.g. the clock widget provides a large circle for the face, twelve circles for each number and 3 lines for each clock hand. If this was created without the widget the programmers would need to draw 13 circles and 3 hands in separate primitive commands.

There are currently 14 in-built widgets and the sample functions are in the form of "SAMAPP_CoPro_Widget_xxx".

- SAMAPP_CoPro_Widget_Logo();
- SAMAPP_CoPro_Widget_Text();
- SAMAPP_CoPro_Widget_Number();
- SAMAPP_CoPro_Widget_Button();
- SAMAPP_CoPro_Widget_Clock();
- SAMAPP_CoPro_Widget_Guage();
- SAMAPP_CoPro_Widget_Gradient();
- SAMAPP_CoPro_Widget_Keys();
- SAMAPP_CoPro_Widget_Progressbar();
- SAMAPP_CoPro_Widget_Scroll();
- SAMAPP_CoPro_Widget_Slider();
- SAMAPP_CoPro_Widget_Dial();
- SAMAPP_CoPro_Widget_Toggle();
- SAMAPP_CoPro_Widget_Spinner();

The following functions are designed to demonstrate additional FT800 commands, which are frequently used by programmers to simplify a project. They are in the form of "SAMAPP_CoPro_xxx".

/*Screen calibrate example*/
- SAMAPP_CoPro_Calibrate();
- SAMAPP_CoPro_Screensaver();

/*Matrix example for Bitmap manipulation*/
- SAMAPP_CoPro_Matrix();

/*Appending block of memory to the current display list*/
- SAMAPP_CoPro_AppendCmds();

/*Decompress functionality example*/
- SAMAPP_CoPro_Inflate();

/*JPEG decoding functionality example*/
- SAMAPP_CoPro_Loadimage();

/*Customer Font example*/
- SAMAPP_CoPro_Setfont();

/*Track usage example for touch*/
- SAMAPP_CoPro_Track();

/*Screenshot example*/
- SAMAPP_CoPro_Snapshot();

/*Sketch example*/
- SAMAPP_CoPro_Sketch();

#### 2.2.5.3 Audio & Touch group

/* Audio playback API */

- SAMAPP_Aud_Music_Player();

/* Audio Playback sample function in streaming way*/

- SAMAPP_Aud_Music_Player_Streaming();

/*FT800 Built-In Sound sample function*/

- SAMAPP_Sound()

/*FT800 Touch and Tag usage sample function*/

- SAMAPP_Touch()

/* FT800 Track coprocessor engine command usage sample */

- SAMAPP_CoPro_Track();

/* FT800 keys widget and touch tag example*/

- SAMAPP_CoPro_Widget_Keys_Interactive();

#### 2.2.5.4 Host Command Group

/*Toggle the PD_N pin of FT800 for power cycle*/

- Ft_Gpu_Hal_Powercycle ()

/*
FT800 Host Command Function: users can send the respective host commands to achieve clock source selection, power mode switch, frequency selection as well as core reset.
*/

- Ft_Gpu_HostCommand()

/*
This API defines 6 scenarios of power mode switch, implemented by calling functions above.
*/

- SAMAPP_PowerMode()

## 3 Helpful Hints

- Audio playback functions "SAMAPP_Aud_Music_Player" and "SAMAPP_Aud_Music_Player_Streaming()", are not available on the Arduino Platform due to the audio playback file being too large to fit into the ATMega328 flash.

- Note that a calibration procedure (e.g. `SAMAPP_CoPro_Calibrate();`) is required if experimenting with the touch screen feature.

## 4 4D- Systems

4D Systems designs and manufactures compact and cost effective Intelligent Display Modules using the latest state of the art OLED and LCD technology with an embedded custom graphics processor. To find out more about 4D Systems and more importantly the FT800 support they provide, visit their website at www.4dsystems.com.au

# 5 Contact Information

### Head Office – Glasgow, UK

Future Technology Devices International Limited
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

| | |
|---|---|
| E-mail (Sales) | sales1@ftdichip.com |
| E-mail (Support) | support1@ftdichip.com |
| E-mail (General Enquiries) | admin1@ftdichip.com |

### Branch Office – Tigard, Oregon, USA

Future Technology Devices International Limited
(USA)
7130 SW Fir Loop
Tigard, OR 97223-8160
USA
Tel: +1 (503) 547 0988
Fax: +1 (503) 547 0987

| | |
|---|---|
| E-Mail (Sales) | us.sales@ftdichip.com |
| E-Mail (Support) | us.support@ftdichip.com |
| E-Mail (General Enquiries) | us.admin@ftdichip.com |

### Branch Office – Taipei, Taiwan

Future Technology Devices International Limited
(Taiwan)
2F, No. 516, Sec. 1, NeiHu Road
Taipei 114
Taiwan , R.O.C.
Tel: +886 (0) 2 8791 3570
Fax: +886 (0) 2 8791 3576

| | |
|---|---|
| E-mail (Sales) | tw.sales1@ftdichip.com |
| E-mail (Support) | tw.support1@ftdichip.com |
| E-mail (General Enquiries) | tw.admin1@ftdichip.com |

### Branch Office – Shanghai, China

Future Technology Devices International Limited
(China)
Room 1103, No. 666 West Huaihai Road,
Shanghai, 200052
China
Tel: +86 21 62351596
Fax: +86 21 62351595

| | |
|---|---|
| E-mail (Sales) | cn.sales@ftdichip.com |
| E-mail (Support) | cn.support@ftdichip.com |
| E-mail (General Enquiries) | cn.admin@ftdichip.com |

### Web Site

www.ftdichip.com

# Appendix A – References

## Document References

DS_FT800 FT800 Datasheet

PG_FT800 FT800 Programmer Guide

AN_240 FT800 From the Ground Up

4D Systems (www.4dsystems.com.au)

Arduino (http://arduino.cc/)

## Acronyms and Abbreviations

| Terms | Description |
|---|---|
| EVE | Embedded Video Engine |
| GPIO | General Purpose Input / Output |
| $I^2C$ | Inter-Integrated Circuit |
| IC | Integrated Circuit |
| MCU | Microcontroller |
| QVGA | Quarter VGA (320 x 240 pixel display size) |
| SPI | Serial Peripheral Interface |
| TFT | Thin-Film Transistor |
| VGA | Video Graphics Array |
| WQVGA | Wide Quarter VGA (480 x 272 pixel display size) |
| FPC | Flat Panel Cable |

# Appendix B – List of Tables & Figures

## List of Figures

## Appendix C – Revision History

|  |  |
|---|---|
| Document Title: | AN_280 DLCD-FT843 SampleApp Arduino Introduction |
| Document Reference No.: | FT_000943 |
| Clearance No.: | FTDI# 355 |
| Product Page: | http://www.ftdichip.com/EVE.htm |
| Document Feedback: | Send Feedback |

| Revision | Changes | Date |
|---|---|---|
| draft | Initial Release | 2013-09-30 |
| 1.0 | First Release | 2013-10-30 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |