AN_246

VM800 Series 'Sample App'

Arduino Introduction

Document Reference No.:FT_000848

Version 1.0

Issue Date: 14 August 2013

This document introduces how to setup the FT800 Sample Application running on an Arduino Pro 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 2
1.1 Audience 2
1.2 Scope 2
1.3 Overview 3
1.3.1 Hardware 3
1.3.2 Application flow 4
1.3.3 Architecture 5
1.4 Hardware requirement 5
1.5 Software requirement 5
1.5.1 Software package introduction 5
2 Set up steps 7
2.1 Hardware Connection 7
2.2 Source code build and download 7
2.2.1 Determine the screen size 8
2.2.2 Determine the group of functions to build and run 8
2.2.3 Source file brief 9
2.2.4 Project file brief 9
2.2.5 Major function groups in sample application 9
3 Helpful Hints 12
4 Contact Information 13
5 Appendix A– References 14
Document References 14
Acronyms and Abbreviations 14
6 Appendix B – List of Tables & Figures 15
7 Appendix C– Revision History 16

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^2C 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 the Arduino Pro platform.

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 Arduino Pro, 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 Arduino Pro would be necessary for other MCU's.

This document introduces how to set up and use the Sample Application with an FTDI VM800 series development system in conjunction with an Arduino Pro platform. At the time of writing the series includes VM800B, VM800C and VM800P development systems. Further information regarding the FT800 programming language or pseudo-code can be found in the FT800 Programmer Guide.

For VM800B, VM800C and VM800P development board details, please refer to the datasheets: DS_VM800C or DS_VM800B or DS_VM800P. To learn more about Arduino Pro and its IDE, please check http://www.arduino.cc

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

We recommend to use the Arduino Pro based on ATmeg328.

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 Pro Platform, knowledge of Arduino Pro hardware and IDE (Independent Design Environment) is required.

1.2 Scope

The Sample Application mentioned in this document is created in the Arduino Pro IDE and runs on Arduino Pro connected to a VM800B or VM800C.

The VM800P module has an integral Arduino Pro processor built in.

The Sample Application project comprises the source code as well as project files.

1.3 Overview

1.3.1 Hardware

The diagram below gives the basic hardware setup.

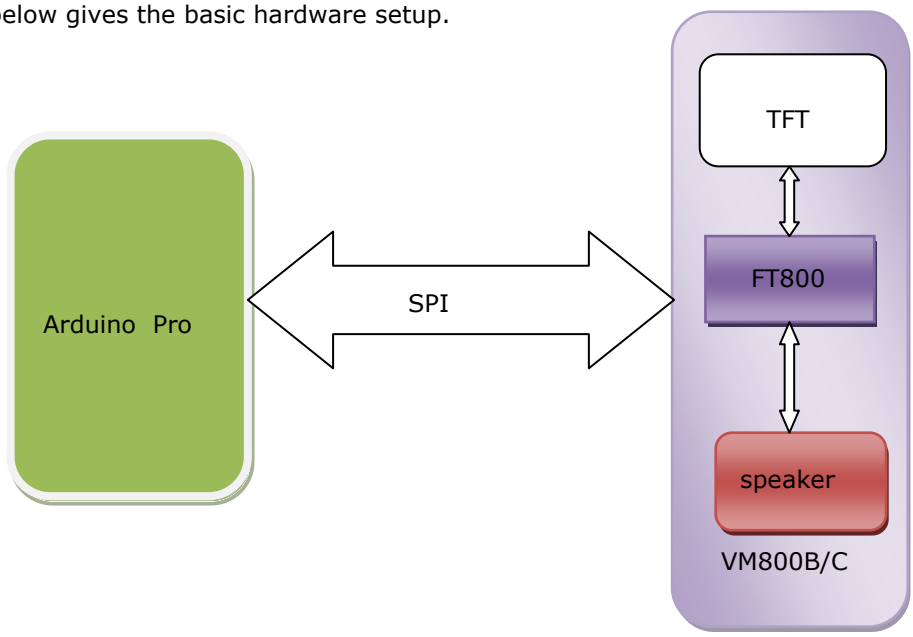

Figure 1.1: Block Diagram of Setup

1.3.2 Application flow

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

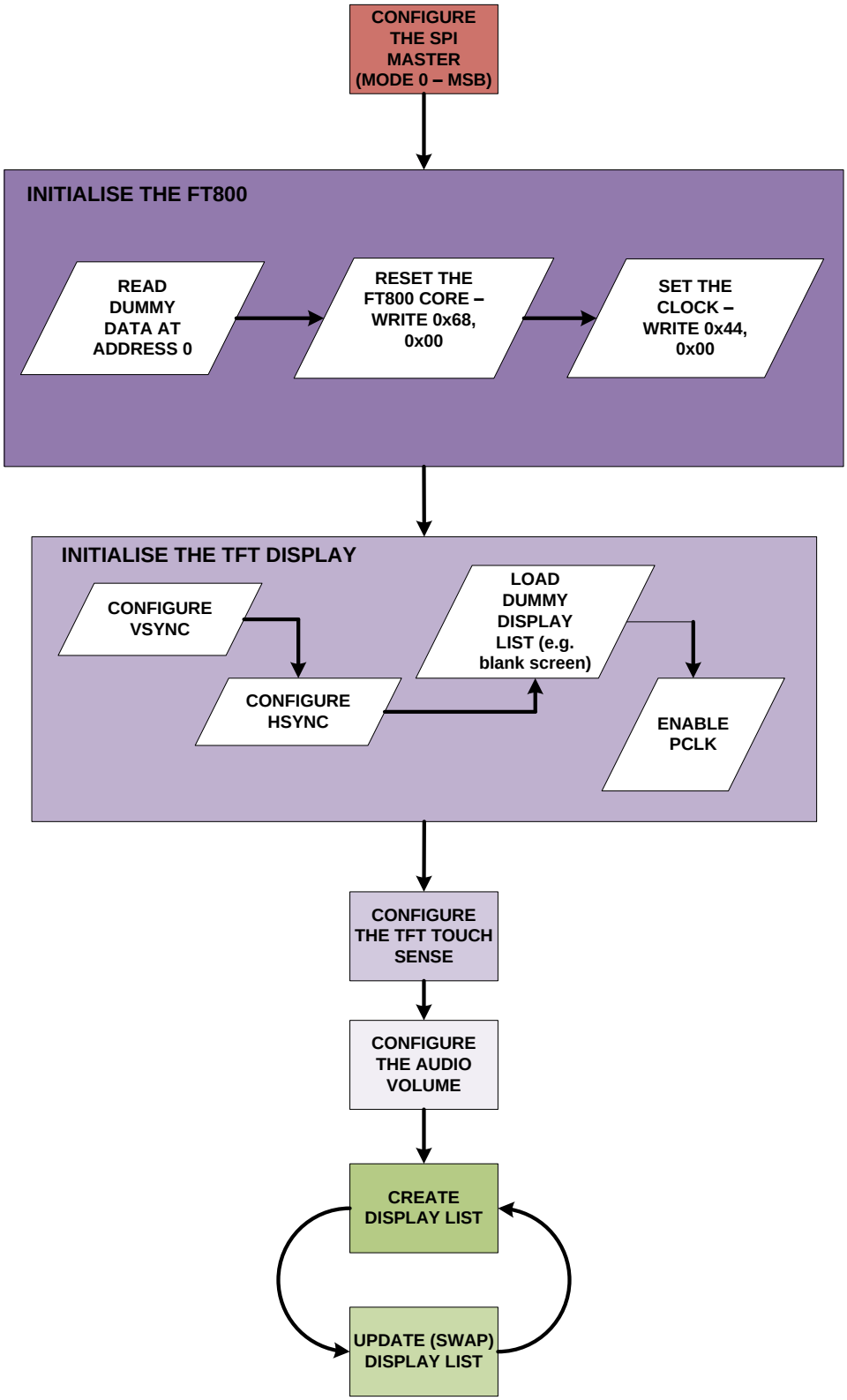

Figure 1.2: Application Flow

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.

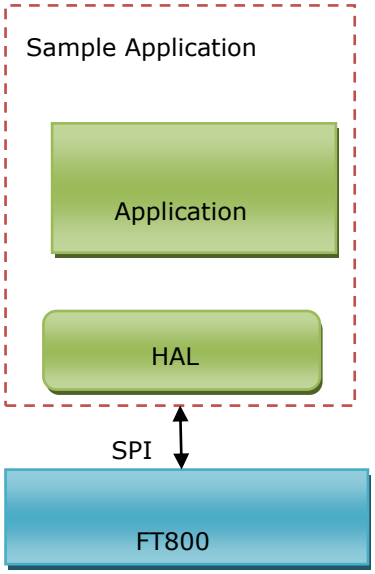

Figure 1.3: Architecture diagram

1.4 Hardware requirement

- VM800B, VM800C or VM800P development kit.

  Note that the development system has an option for 3 different size displays, 3.5”, 4.3”, or 5.0”(VM800C35A-xx, VM800B35A-xx, VM800P35A-xx, VM800C43A-xx, VM800B43A-xx, VM800P43A-xx VM800C50A-xx, VM800B50A-xx, VM800P50A-xx respectively).
- One USB cable with a MicroB connector to provide power to the VM800 board. It is optional if battery is provided for power supply.

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".
- Folder "Docs\Arduino" contains this document.
- The other folders are not relevant to the Arduino Pro 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 table below describes the connection between the Arduino Pro and the VM800 board. Please check the datasheet of the VM800 system for more details.

For details of the power supply requirements of Arduino Pro and VM800, check the Arduino Pro datasheet and VM800 system datasheet accordingly.

Arduino Pro	VM800C Board	Description
Digital Pin 13	SCK in J5	Signal SCK of SPI
Digital Pin 11	MOSI in J5	Signal MOSI of SPI
Digital Pin 12	MISO in J5	Signal MISO of SPI
Digital Pin 10	CS# in J5	Signal CS of SPI
GND	GND in J5	Ground
Digital Pin 4	PD# in J5	Power Down Pin(PD_N#) of FT800
Digital Pin 3	INT# in J5	Interrupt Pin(INT_N#) of FT800

Table 1: Hardware signal connection

Note: The Arduino Pro is already part of the VM800P module and no external connections are required.

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:

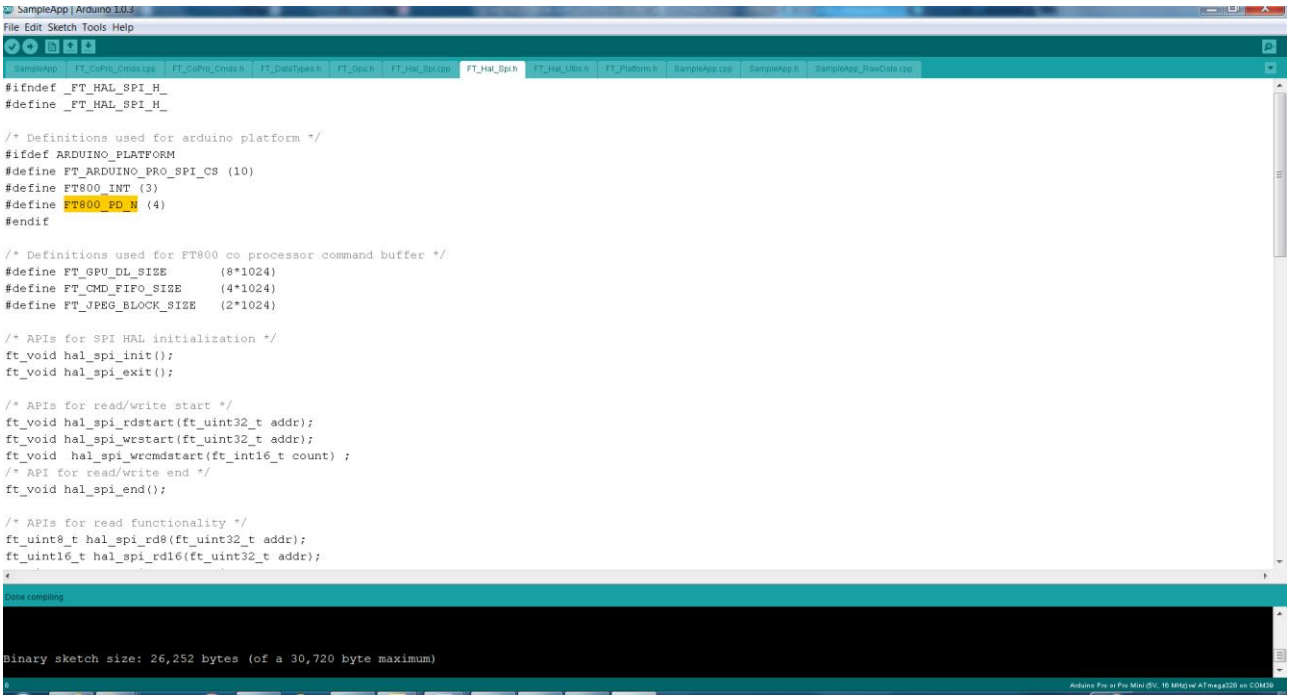

Figure 1.4 :Arduino IDE with the opened project

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

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

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 the 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 Arduino Pro flash size(32KBytes), it is impossible to put all the functions together and program them into the Arduino Pro. 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 Arduino Pro flash.

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

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 the 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*/
SAMAPP_GPU_Points();
/*draw a triangle*/
SAMAPP_Gpu_Polygon();
/*draw lines*/
SAMAPP_GPU_Lines();
/*draw rectangles*/
SAMAPP_GPU_Rectangles();
/*draw bitmaps*/
SAMAPP_GPU_Bitmap();
/*draws chars with different fonts*/
SAMAPP_GPU_Fonts();
```

- SAMAPP_GPU_Text8x8();
- SAMAPP_GPU_TextVGA();

  /*draws a bargraph*/
- SAMAPP_GPU_Bargraph();
- SAMAPP_GPU_LineStrips();
- SAMAPP_GPU_EdgeStrips();

  /*example of cutting away an active area on the display*/
- SAMAPP_GPU_Scissor();

  /*Font and Points Primitives combination*/
- SAMAPP_GPU_FtdiString();

  /*Call and Return Primitives combination*/
- SAMAPP_GPU_StreetMap();

  /*Additive blending of fonts*/
- SAMAPP_GPU_AdditiveBlendText();

  /*Usage of Macro*/
- SAMAPP_GPU_MacroUsage();

  /*Additive blending of points*/
- 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 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 Pro Platform due to the audio playback file being too large to fit into the Arduino Pro flash.

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

4 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
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) asia.sales1@ftdichip.com
E-mail (Support) asia.support1@ftdichip.com
E-mail (General Enquiries) asia.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

http://ftdichip.com

5 Appendix A– References

Document References

1. datasheet for VM800C: DS_VM800C EVE
2. datasheet for VM800B: DS_VM800B EVE
3. FTDI MPSSE for SPI application note
4. FT800 Programmer Guide
5. FT800 Embedded Video Engine Datasheet

Acronyms and Abbreviations

Terms	Description
Arduino Pro	The open source platform variety based on ATMEL's ATMEGA chipset
EVE	Embedded Video Engine
SPI	Serial Peripheral Interface
UI	User Interface
USB	Universal Serial Bus

6 Appendix B – List of Tables & Figures

List of Figures and Tables

Figure 1.1: Block Diagram of Setup 3
Figure 1.2: Application Flow 4
Figure 1.3: Architecture diagram 5
Table 1: Hardware signal connection 7
Figure 1.4 :Arduino IDE with the opened project 8

7 Appendix C– Revision History

Document Title:	AN_246 VM800C/B Sample App Arduino Introduction
Document Reference No.:	FT_000848
Clearance No.:	FTDI# 389
Product Page:	http://www.ftdichip.com/EVE.htm
Document Feedback:	Send Feedback

Revision	Changes	Date
1.0	Initial Release	2013-06-06