Application Note

AN_381

ME810A HV35R Sample Application

Version 1.0

Issue Date: 2018-01-08

This application note introduces how to setup the FT8XX Sample Application running on a FT9XX Series MCU, MSVC and Arduino. The objective of the Sample Application is to enable users to become familiar with the usage of FT8XX, the design flow and display list used to design the desired user interface or visual effect.

Table of Contents

1 Introduction 4
1.1 Audience 5
1.2 Scope 5
2 Overview 6
2.1 Hardware Block Diagram 6
2.2 Application Flow 7
2.3 Architecture 8
2.4 Hardware Requirements 8
2.5 Software Requirements 9
2.5.1 Software package introduction 9
3 Setup 10
3.1 Hardware Connection 10
3.1.1 MM900EV1A 10
3.2 Source Code Build and Download 11
3.2.1 Determine the screen size and choose module 12
3.2.2 Determine the group of functions to build and run 13
3.2.3 Source File Brief 13
3.2.4 Project File Brief 14
3.2.5 Major Function Groups in Sample Application 14
4 Helpful Hints 21
5 ME810A-HV35R 22
6 Contact Information 23
Appendix A – References 24
Document References 24
Acronyms and Abbreviations 24
Appendix B – List of Figures & Tables 25
List of Figures 25

List of Tables.....25

Appendix C – Revision History 26

1 Introduction

The FT8XX combines display, audio and touch functionality into one single chip, powered by Bridgetek's advanced EVE technology (Embedded Video Engine). The FT8XX devices interface with the system MCU either via an SPI or QSPI interface. To enable customers to more easily utilize the functionality of the FT8XX in a project, a Sample Application is provided here for tutorial purposes. The sample application has been written for FT90X MCU, MSVC and Arduino platforms.

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 FT8XX.

Note that although the basic project can be tested with Bridgetek FT90X MCU, the code relating to the creation of the screens could be reused in different MCU design environments. In addition, the set-up steps for the FT90X would need to be ported over to the selected MCU.

This document introduces how to set up and use the Sample Application with a Bridgetek ME800A-HV35R development system (see Figure 1) in conjunction with the FT900 platform. Further information regarding the FT8XX programming language or pseudo-code can be found in the FT8XX Programmer Guide.

For FT90X details please refer to http://brtchip.com/mcu/.
To learn more about FT8XX, please refer http://brtchip.com/eve/.

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

Figure 1 - 3.5 Inch Display with FT8XX

Figure 2 shows the 3.5 inch display connected to the FT90X Board. The FT8XX board enables system designers to rapidly create high quality; human machine interfaces (HMIs). It includes a 3.5 inch TFT display (with 320 x 480 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 3.5 inch Display is connected to the MM900EV via the ME800A Module.

Figure 2 - ME810A-HV35R Connected to MM900EV1A

1.1 Audience

This document assumes that the audiences are familiar with the datasheet and programmer's guide of the FT8XX. In addition, familiarity of the C/C++programming language is necessary to understand the Sample Application source code. To understand the SPI of the FT900 Platform, knowledge of FT900 microcontroller and IDE (Integrated Development Environment) is required.

1.2 Scope

The Sample Application mentioned in this document is created in Eclipse for FT900 and runs on FT900 board connected to the FT8XX with Display module. It is comprised of the source code as well as project files.

2 Overview

2.1 Hardware Block Diagram

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

The MM900EV1A module (shown in green block) is connected to ME810A-HV35R via QSPI; LCD is connected to ME810A-HV35R module. The SampleApp code includes audio, which can be played by Buzzer on the ME810A-HV35R module or audio codec on MM900EVxA module.

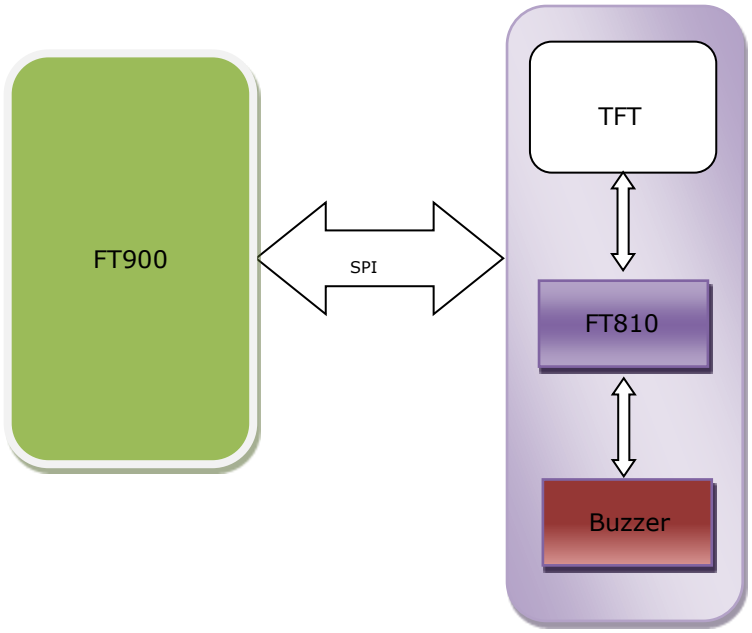

Figure 3 - Hardware Setup Block Diagram

2.2 Application Flow

The diagram below gives the basic flow and structure for configuring the FT810 in an application.

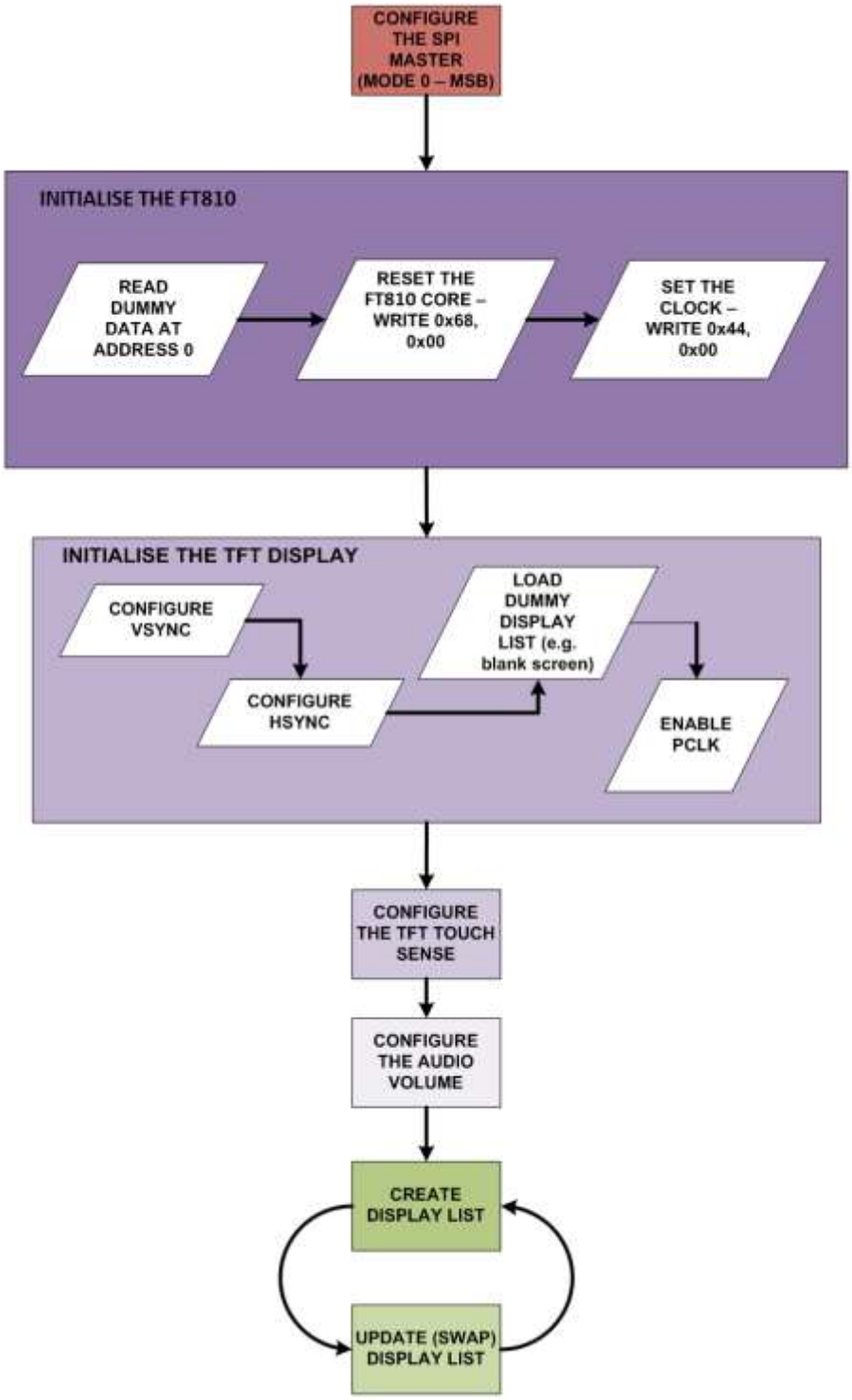

Figure 4 - Application Flow

2.3 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.brtchip.com for the latest support.

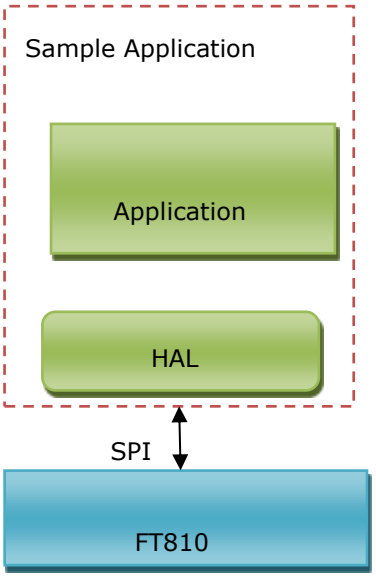

Figure 5 – Architecture Diagram

2.4 Hardware Requirements

- ME810A-HV35R module for FT810.
- UMFTPD2A Module for program/debug.
- FT900 board: MM900EV1A/MM900EV2A/MM900EV3A/MM900EV-LITE.
- Micro USB cables.

2.5 Software Requirements

- FT900 Toolchain
- FT810 Sample Application release package.

2.5.1 Software package introduction

2.5.1.1 Folder Introduction

- Folder "Project\FT90x" contains the project file.
- The source files are included in Src folder.
- The header files are included in Hdr and Hdr\FT90x folder.
- The library files such as fatfs are included in bin\FT90x\

2.5.1.2 Dependency

The Sample Application uses the SPI, UART etc. libraries provided by Bridgetek as part of the FT900 Toolchain.

3 Setup

3.1 Hardware Connection

The MM900EV1A module is mated onto the top of the ME810A-HV35R Board as shown in the below figure**.** The 3.5 inch DISPLAY panel cable connects into the FPC socket on the top of the ME module. USB micro cables are used to power both boards. UMFTPD2A Module is used for downloading the program to the FT900 Board.

Figure 6 – Hardware Connections

3.1.1 MM900EV1A

Figure 9 shows the MM900EV1A module. The MM900EV development modules provide full hardware feature support for the FT90X processor in a variety of form factors. The modules also provide a connector for accessing external plug-in hardware over an SPI interface. There are three module types available: MM900EV1A, which comes without a camera, MM900EV2A, with an inbuilt front facing CMOS camera, and MM900EV3A, with a back facing camera.

Key features:

• Ethernet 10/100Base-T interface, RJ45 MAGJACK connector with 2 LED status indication

• Independent USB Hi-speed device and host port

• Built in small and low noise microphone module

• 3.5mm audio jack for stereo audio output with a mono microphone input

• Debugger interface for FT900 -ash programming and EFUSE configuration

• RGB LED with 24 bit color

• Micro SD card socket supporting SD3.0 specification

• SPI/QSPI interface exposed over a 16-pin header connector to connect with accessory cards such as the ME800A-HV35R 3.5" resistive touch HVGA display daughter board

• 40 pins double row header for extending IOs

- 5V power jack source
- Built in high quality CMOS Camera module. (MM900EV2A and MM900EV3A only)

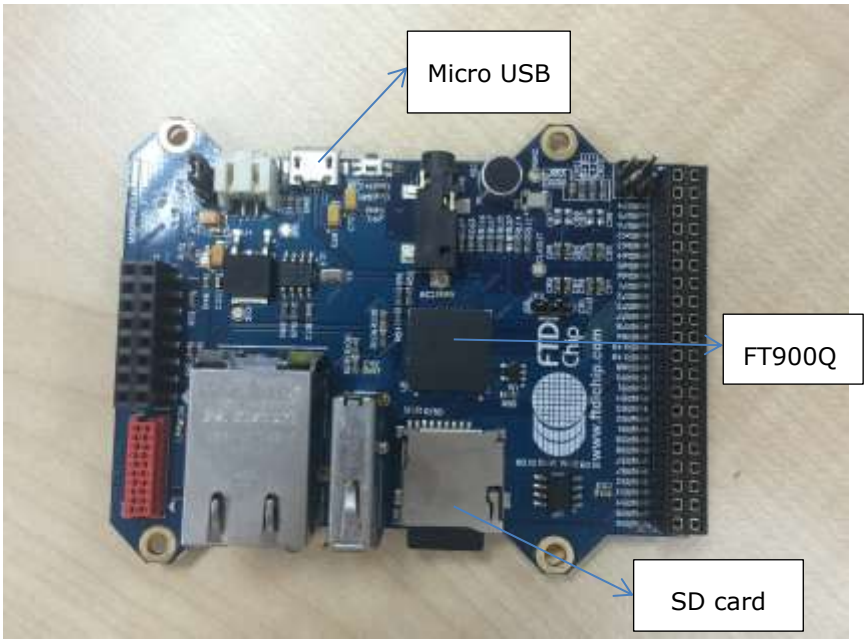

Figure 7 – MM900EV Module

3.2 Source Code Build and Download

To build the project, import the project file from "Project\FT90x\SampleApp\" into the Eclipse IDE and the following screen will be shown:

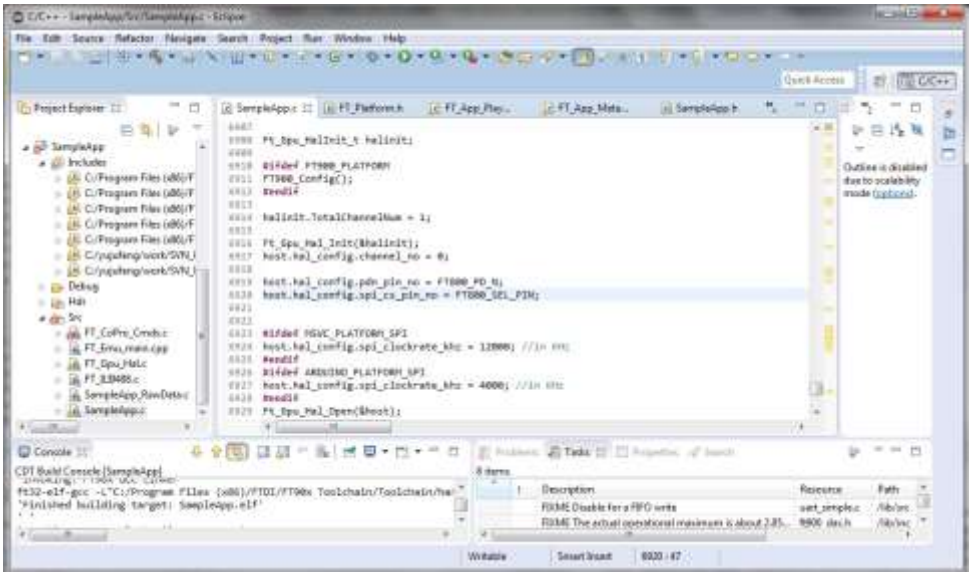

Figure 8 – Eclipse IDE with the Opened Project

Note: When importing into Eclipse, do not select 'Copy projects into workspace' as this does not copy all files.

Right-click the project and select "Build project" to build the code and use "FT900Prog GUI Launcher" to download the binary generated into the FT900 memory. The FT900Prog GUI Launcher is installed together with Eclipse for FT900.

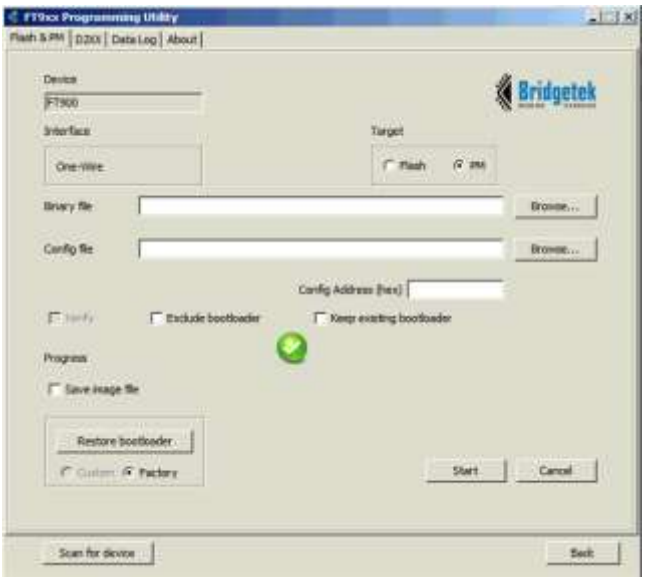

Figure 9 – FT900Prog GUI Launcher

For further details of Eclipse usage, see AN_325 FT9XX Toolchain Installation Guide.

3.2.1 Determine the screen size and choose module

The following settings are available in \Hdr\FT90x\ Platform.h.

Switch the compilation macro for different modules:

```
#define ME800A_HV35R
#define ME810A_HV35R
#define ME813A_WV7C
```

For 3.5 inch displays, ensure the compilation macro switch "#define ME800A_HV35R" or "#define ME810A_HV35R" is uncommented. Only one module shall be selected (un-commented) at a time.

For other development boards (4.3" and 5.0" displays), the macro selection should be adjusted to ensure the correct display resolution and size.

For a more customized setting, the user can comment out *all* of the module-specific defines above and select the individual options below.

```
#if (!defined(ME800A_HV35R) && !defined(ME810A_HV35R) && !defined(ME813A_WV7C))
/* platform specific macros */
#define FT900_PLATFORM                                  (1)
/* Display configuration specific macros */
#define DISPLAY_RESOLUTION_QVGA                         (1)
#define DISPLAY_RESOLUTION_WQVGA                        (1)
#define DISPLAY_RESOLUTION_WVGA                         (1)
#define DISPLAY_RESOLUTION_HVGA_PORTRAIT                (1)

/* Chip configuration specific macros */
#define FT800_ENABLE                                    (1)
```

```
#define FT801_ENABLE                                                    (1)
#define FT810_ENABLE                                                    (1)
#define FT811_ENABLE                                                    (1)
#define FT812_ENABLE                                                    (1)
#define FT813_ENABLE                                                    (1)

/* SPI specific macros - compile time switches for SPI single, dial and quad use
cases */
#define ENABLE_SPI_SINGLE                                               (1)
#define ENABLE_SPI_DUAL                                                 (1)
#define ENABLE_SPI_QUAD                                                 (1)

/* Display driver configurations - mainly for ME900EV1 modules */
#define ENABLE_ILI9488_HVGA_PORTRAIT                        (1)

#endif /* #if (!defined(ME800A) && !defined(ME813A)) */
```

Upon selection, re-build the project so that the code builds with the selected options.

3.2.2 Determine the group of functions to build and run

The demonstrations are split into sets which can be selected via #defines.

```
#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
#define SAMAPP_ENABLE_APIS_SET5
#define SAMAPP_ENABLE_APIS_SET6
#define SAMAPP_ENABLE_APIS_SET7
#define SAMAPP_ENABLE_APIS_SET8
#define SAMAPP_ENABLE_APIS_SET9
```

Please enable only one API set at a time to ensure that the binary build will fit into the flash. These can be found in Hdr/SampleApp.h.

Note the categories mentioned here are not same as the groups mentioned in Section3.2.5. The groups here are defined for the purpose of running on the FT900 platform.

Note: When rebuilding the code with changes made to this header file, right-click on the project and select 'Clean Project' before building the project.

3.2.3 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".

"Gpu_Hal.cpp" defines the transportation layer functions, which provides one SPI abstraction layer to access the FT8XX. Editing the file allows for porting the application to alternative MCU's and compilers with minimal effort. It is more specific to the SPI master interface.

"CoPro_Cmds.cpp" defines the APIs of the FT8XX coprocessor engine commands. This file is structured to be generic and could be ported to other projects for other target MCU's.

"GPU.h" defines the FT8XX specific instruction parameters, register names and memory maps. The contents of this file relate directly to the FT8XX 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.

"ILI9488.c" defines the APIs for configuring ILI9488 display driver. ILI9488 is connected in 4 wire mode and bit bang approach is utilized for programming the display driver.

3.2.4 Project File Brief

The ".project " and ".cproject" are the project files used by the Eclipse IDE and include all the necessary source files for the project.

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

3.2.5 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.

```
SAMAPP_ENABLE_APIS_SET0
	SAMAPP_GPU_Points();
	SAMAPP_GPU_Lines();
	SAMAPP_GPU_Rectangles();
	SAMAPP_GPU_Bitmap();
#if defined(MSVC_PLATFORM) || defined(MSVC_FT800EMU)
	SAMAPP_GPU_Bitmap_Palette();
	SAMAPP_GPU_Bitmap_Palette_Background();
#endif
	SAMAPP_GPU_Fonts();
	SAMAPP_GPU_Text8x8();
	SAMAPP_GPU_TextVGA();
	SAMAPP_GPU_Bargraph();
	SAMAPP_GPU_LineStrips();
	SAMAPP_GPU_EdgeStrips();
	SAMAPP_GPU_Scissor();
	SAMAPP_Gpu_Polygon();
	SAMAPP_Gpu_Cube();
	SAMAPP_GPU_Ball_Stencil();
	SAMAPP_GPU_String();
	SAMAPP_GPU_StreetMap();
	SAMAPP_GPU_AdditiveBlendText();
	SAMAPP_GPU_MacroUsage();
	SAMAPP_GPU_AdditiveBlendPoints();
	SAMAPP_API_Screen("Set0   End!");

SAMAPP_ENABLE_APIS_SET1
	SAMAPP_CoPro_Logo();
	SAMAPP_CoPro_Widget_Calibrate();
	SAMAPP_CoPro_Widget_Clock();
	SAMAPP_CoPro_Widget_Guage();
	SAMAPP_CoPro_Widget_Gradient();
	SAMAPP_CoPro_Widget_Keys();
	SAMAPP_CoPro_Widget_Keys_Interactive();
```

```
        SAMAPP_CoPro_Widget_Progressbar();
        SAMAPP_CoPro_Widget_Scroll();
        SAMAPP_CoPro_Widget_Slider();
        SAMAPP_CoPro_Widget_Dial();
        SAMAPP_CoPro_Widget_Toggle();
        SAMAPP_CoPro_Widget_Spinner();
        SAMAPP_PowerMode();
        SAMAPP_CoPro_Widget_Calibrate();
        SAMAPP_Touch();


SAMAPP_ENABLE_APIS_SET2
        SAMAPP_CoPro_Inflate();
        SAMAPP_CoPro_Loadimage();

SAMAPP_ENABLE_APIS_SET3
        SAMAPP_CoPro_Setfont();

SAMAPP_ENABLE_APIS_SET4
        /* Sample code for coprocessor widgets */
        SAMAPP_CoPro_Widget_Text();
        SAMAPP_CoPro_Widget_Number();
        SAMAPP_CoPro_Widget_Button();
        SAMAPP_CoPro_AppendCmds();
        SAMAPP_CoPro_Widget_Calibrate();

        Gpu_Hal_Wr8(phost, REG_VOL_SOUND, 255);
        SAMAPP_Sound();
        SAMAPP_CoPro_Screensaver();
        SAMAPP_CoPro_Snapshot();
        SAMAPP_CoPro_Sketch();
        SAMAPP_CoPro_Matrix();
        SAMAPP_CoPro_Track();
#if defined(MSVC_PLATFORM) || defined(MSVC_FT800EMU)
        //Enable Audio out by setting GPIO
        Gpu_Hal_Wr8(phost, REG_GPIO,0x083 | Gpu_Hal_Rd8(phost,REG_GPIO));
        /* Audio playback api*/
        Gpu_Hal_Wr8(phost, REG_VOL_SOUND,255);
        SAMAPP_Aud_Music_Player_Streaming();
#endif


SAMAPP_ENABLE_APIS_SET5
#if defined(MSVC_PLATFORM) || defined(MSVC_FT800EMU) || defined(FT900_PLATFORM)
        SAMAPP_ChineseFont();
#endif

SAMAPP_ENABLE_APIS_SET6
#if defined(MSVC_PLATFORM) || defined(MSVC_FT800EMU)
#if defined(FT801_ENABLE) || defined(FT811_ENABLE)

        SAMAPP_CoPro_Widget_Calibrate();
        SAMAPP_API_Screen("Main Windows");
        MainWindow();
#endif
```

```
#endif

SAMAPP_ENABLE_APIS_SET7
#if defined(FT801_ENABLE) || defined(FT811_ENABLE)
	SAMAPP_CoPro_Widget_Calibrate();
	SAMAPP_API_Screen("Bouncing Squares");
	BouncingSquares();
	SAMAPP_API_Screen("Bouncing Circles");
	BouncingCircles();
	SAMAPP_API_Screen("Bouncing Points");
	BouncingPoints();
	SAMAPP_API_Screen("Moving Points");
	MovingPoints();
#endif

SAMAPP_ENABLE_APIS_SET8
#if defined(FT81X_ENABLE)
	SAMAPP_CoPro_Widget_Calibrate();
	Gpu_CoCmd_ColdStart(phost);
	SAMAPP_API_Screen("Multi-track on an object.");
	SAMAPP_CoPro_MultiTracker();
	SAMAPP_API_Screen("Number Bases");
	NumberBases();

	SAMAPP_API_Screen("New bitmap formats, options, and sizes.");
	FT81xBitmapFormatAndOptions();  //png loading
	SAMAPP_API_Screen("Screen Orientation");
	ScreenOrientation();
	SAMAPP_API_Screen("Bitmap up to 2048pix x 2048pix.");
	HigherResolutionBitmap();
	SAMAPP_API_Screen("Paletted bitmap.");
	SAMAPP_81X_Paletted_Bitmap();
	SAMAPP_API_Screen("DXT1: reduce bitmap size upto 4 folds.");
	SAMAPP_GPU_DXT1();


#if !defined(FT900_PLATFORM) && (!defined(ARDUINO_PLATFORM))
	{
		SAMAPP_API_Screen("AVI video playback");
		VideoPlayback();

		SAMAPP_API_Screen("AVI video playback via CMD buffer");
		VideoPlaybackViaCMDBuff();

		SAMAPP_API_Screen("AVI video playback via frame by frame");
		VideoPlayBackFrameByFrame();
	}
#endif
#endif
HAL LibraryDemo app porting (6)

SAMAPP_ENABLE_APIS_SET9 (Not available for ft900 platform)
#if defined(FT81X_ENABLE) && !defined(MSVC_FT800EMU)
	SAMAPP_API_Screen("Configurable clock frequency.");
	SAMAPP_ChangeFreq();
```

```
        SAMAPP_81X_PowerOffComponents();
        SAMAPP_API_Screen("Configurable GPIO/IO drive strength..");
        SAMAPP_81X_ChangePadDriveStrength();
#endif
```

3.2.5.1 Primitives Group

The functions in this group are designed to demonstrate the usage of FT8XX primitives.

An FT8XX 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 FT8XX 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_String();
/*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();
```

3.2.5.2 Widgets Group

The functions in this group are designed to demonstrate the FT8XX 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 and calculate the angles etc.

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 FT8XX commands, which are frequently used by programmers to simplify a project. They are in the form of "SAMAPP_CoPro_xxx".

```
/*Screen calibrate example*/
o SAMAPP_CoPro_Calibrate();
o SAMAPP_CoPro_Screensaver();
/*Matrix example for Bitmap manipulation*/
o SAMAPP_CoPro_Matrix();
/*Appending block of memory to the current display list*/
o SAMAPP_CoPro_AppendCmds();
/*Decompress functionality example*/
o SAMAPP_CoPro_Inflate();
/*JPEG decoding functionality example*/
o SAMAPP_CoPro_Loadimage();
/*Customer Font example*/
o SAMAPP_CoPro_Setfont();
/*Track usage example for touch*/
o SAMAPP_CoPro_Track();
/*Screenshot example*/
o SAMAPP_CoPro_Snapshot();
/*Sketch example*/
o SAMAPP_CoPro_Sketch();
```

3.2.5.3 Audio & Touch Group

```
/* Audio playback API */
o SAMAPP_Aud_Music_Player();
/* Audio Playback sample function in streaming way*/
o SAMAPP_Aud_Music_Player_Streaming();
/*FT800 Built-In Sound sample function*/
o 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();

3.2.5.4 Host Command Group

/*Toggle the PD_N pin of FT8xx for power cycle*/

- 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.
*/

- Gpu_HostCommand()

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

- SAMAPP_PowerMode()

4 Helpful Hints

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

All of the APIs that need assets in SD card/storage are disabled. Users can include them by copying content from .\Test folder into SD card and enabling the respective sets in the .\Hdr\SampleApp.h file.

5 ME810A-HV35R

The ME810A-HV35R provides a display, audio and touch HMI companion for the MM900EV. The module includes Bridgetek's FT810 Embedded Video Engine with a 3.5" 320 x 480 HVGA display and resistive touch. An integrated audio Buzzer and LCD backlight control are also included. The 16-pin header connector provides the SPI interface to the MM900EV.

Figure 10 - ME810A-HV35R

6 Contact Information

Head Quarters – Singapore

Bridgetek Pte Ltd
178 Paya Lebar Road, #07-03
Singapore 409030
Tel: +65 6547 4827
Fax: +65 6841 6071

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Branch Office – Taipei, Taiwan

Bridgetek Pte Ltd, Taiwan Branch
2 Floor, No. 516, Sec. 1, Nei Hu Road, Nei Hu District
Taipei 114
Taiwan , R.O.C.
Tel: +886 (2) 8797 5691
Fax: +886 (2) 8751 9737

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Branch Office - Glasgow, United Kingdom

Bridgetek Pte. Ltd.
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) sales.emea@brtichip.com
E-mail (Support) support.emea@brtchip.com

Branch Office – Vietnam

Bridgetek VietNam Company Limited
Lutaco Tower Building, 5th Floor, 173A Nguyen Van Troi,
Ward 11, Phu Nhuan District,
Ho Chi Minh City, Vietnam
Tel : 08 38453222
Fax : 08 38455222

E-mail (Sales) sales.apac@brtchip.com
E-mail (Support) support.apac@brtchip.com

Web Site

http://brtchip.com/

Distributor and Sales Representatives

Please visit the Sales Network page of the Bridgetek Web site for the contact details of our distributor(s) and sales representative(s) in your country.

Appendix A – References

Document References

- AN_391 EVE Platform Guide
- FT8XX Embedded Video Engine Datasheet
- FT8XX Series Programmers Guide
- AN_325 FT9XX Toolchain Installation Guide
- AN_240 FT8XX From the Ground Up
- http://brtchip.com/eve/
- http://brtchip.com/mcu/

Acronyms and Abbreviations

Terms	Description
EVE	Embedded Video Engine
GPIO	General Purpose Input / Output
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
HVGA	Half VGA (320 x 480 pixel display size)

Appendix B – List of Figures & Tables

List of Figures

Figure 1 - 3.5 Inch Display with FT8XX 4
Figure 2 - ME810A-HV35R Connected to MM900EV1A 5
Figure 3 - Hardware Setup Block Diagram 6
Figure 4 - Application Flow 7
Figure 6 – Architecture Diagram 8
Figure 8 – Hardware Connections 10
Figure 9 – MM900EV Module 11
Figure 10 – Eclipse IDE with the Opened Project 11
Figure 11 – FT900Prog GUI Launcher 12
Figure 12 - ME810A-HV35R 22

List of Tables

NA

Appendix C – Revision History

Document Title: AN_381 ME810A HV35R Sample Application
Document Reference No.: BRT_000188
Clearance No.: BRT#123
Product Page: http://brtchip.com/product/
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2018-01-08