Application Note

AN_254

FT800 Designs With Visual TFT

Version 1.0

Issue Date: 2013-09-30

The FTDI FT800 video controller offers a low cost solution for embedded graphics requirements. In addition to the graphics, resistive touch inputs and an audio output provide a complete human machine interface to the outside world.

This application note will describe the process of developing MCU code with MikroElektronika tools to control the FT800.

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 is a peripheral to the main system processor dedicated to rendering graphics, sensing display touch stimulus, and playing audio.

The device is controlled over a low bandwidth SPI or I^2C interface allowing interfacing to nearly any microcontroller with a SPI or I^2C master port. Simple and low-pin-count microcontrollers can now have a high-end graphical user interface by using the FT800.

After selecting the system microcontroller to be used in a design it is important to have the correct tools to develop the firmware running on the microcontroller.

One option is to use compilers from MikroElectronika if your chosen microcontroller is on the list of supported devices. A key benefit in doing this, is that MikroElectronika also provide a graphical interface tool, Visual TFT, that integrates to their compilers.

This application note will demonstrate how the graphical interface of Visual TFT allows users to rapidly develop a display solution by dragging and dropping objects onto the design palette. The majority of actual code is then created with the MikroElectronika libraries at the push of a button to complete the design.

2 VisualTFT FT800 Support

At the time of writing, Visual TFT has FT800 library support for the following processor families:

Stellaris ARM, STM32 ARM, PIC32, ATMEGA.

It is recommended to check with MikroElectronika if a specific device is supported.

This example will use the STM32F207VG processor.

3 Hardware Connection

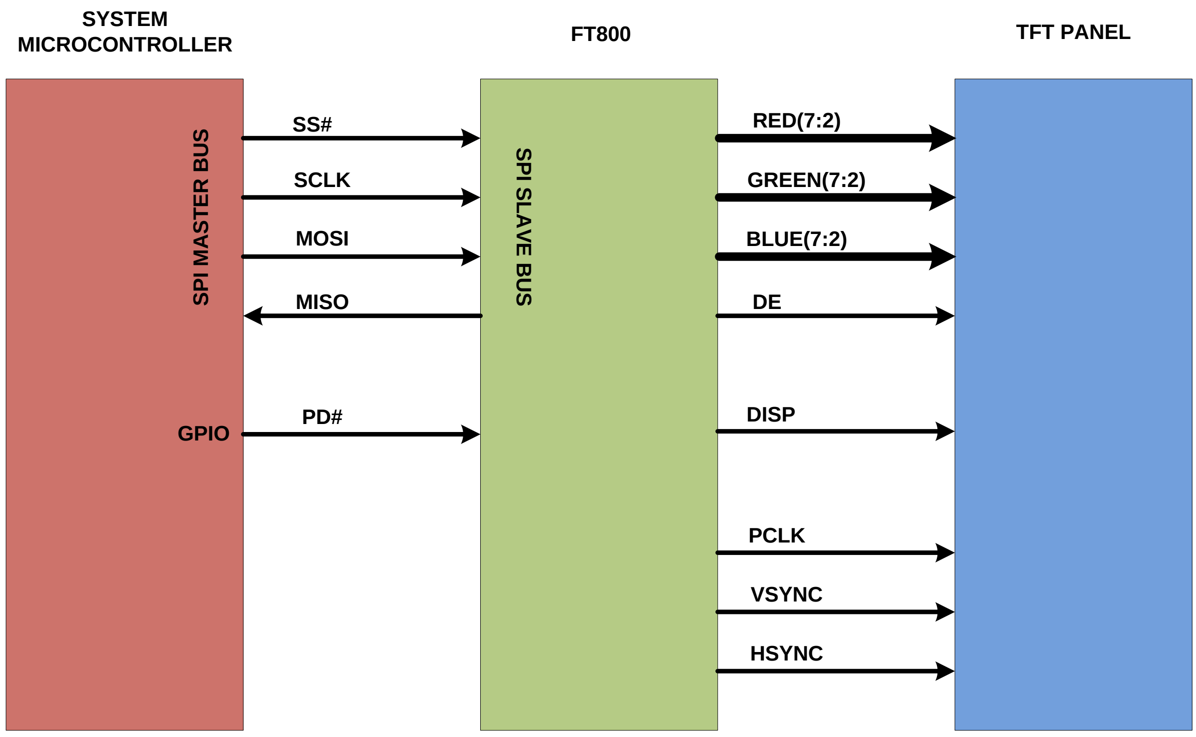


Figure 3.1 Interconnection Diagram

NOTE: The code being developed is running on the system microcontroller. The FT800 does not require code compiled for it. It is simply an SPI peripheral to the system microcontroller.

4 Visual TFT Project

4.1 Creating the project

VisualTFT has a ribbon bar similar to many Microsoft applications. Selecting Project > New will display a pop up where the project name is created and saved.

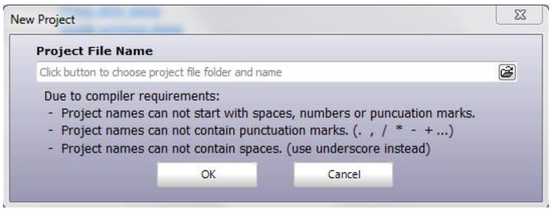


Figure 4.1 New Project

Providing a project name and selecting OK will open the Project Settings window.

4.2 Project Settings

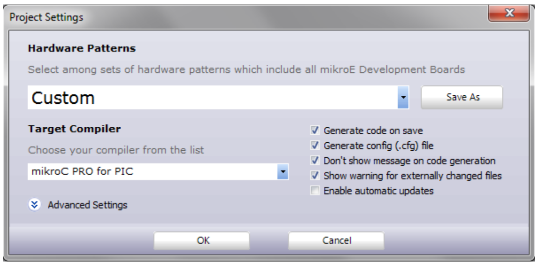
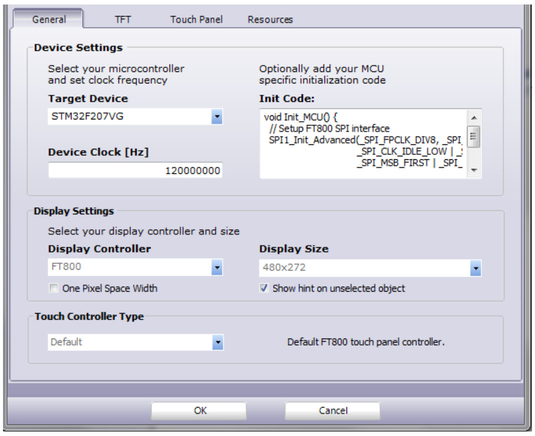


Figure 4.2 Project Settings

The first pull down box (Hardware Patterns) on this screen allows the user to select the processor the application code should target.

The second pull down box (Target Compiler) allows the user to select the compiler that will be used to generate the executable binary code from the object placement performed in Visual TFT.

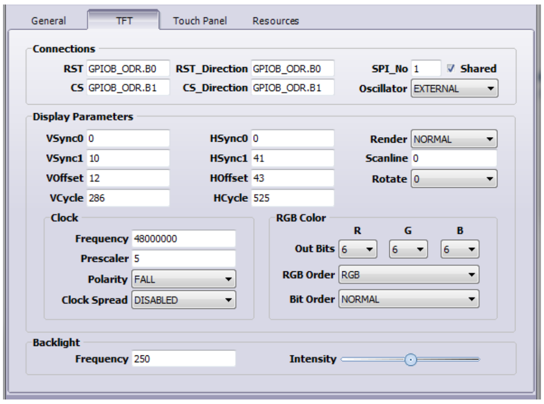
The advanced settings arrow expands to a form that allows the user to create processor specific settings such as clock speed, and initialization of the interface for the SPI/I2C component of the microcontroller.



The example given shows a clock speed of 12MHz with an FT800 display controller connecting to a 480 x 272 display.

Figure 4.3 General Settings

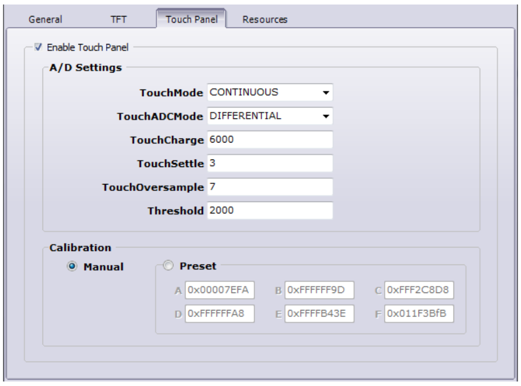
The TFT tab provides a menu for setting basic display parameters. The parameters are defined from the display datasheet.



Polarity and Clock Spread are features the FT800 supports. The polarity setting refers to data latching on the pixel clock while clock spread determines if all RGB signals change at the same time or are staggered on subsequent clock edges to improve noise immunity.

The RST and CS settings define the MCU pins that will drive the FT800 SPI chip select input and reset pins.

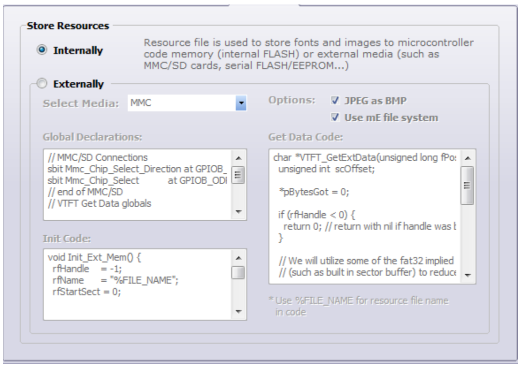
Figure 4.4 TFT Settings



The Touch Panel tab allows the values for the FT800 touch registers to be set. Actual values will be determined from the display datasheet.

For displays without touch capability the values are still written, but have no functional meaning.

Figure 4.5 Touch Panel Settings



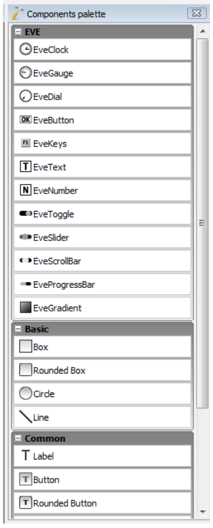
Resource settings are specific to the microcontroller and define whether internal or external memory is used.

Figure 4.6 Resource Settings

When all the settings are made the user can chose to save the configuration as their own template or simply select OK with the defaults.

It is recommended if making edits that the settings are saved with a unique name.

4.3 Placing Objects



Objects to be displayed are dragged from the component palette onto the main window which represents the display as it will be seen in the final project.

Note the palette contains simple shapes such as lines, boxes and circles as well as complex FT800 specific widgets for creating complex objects such as a clock, gauge and slider bars, rapidly.

Figure 4.7 Component Palette

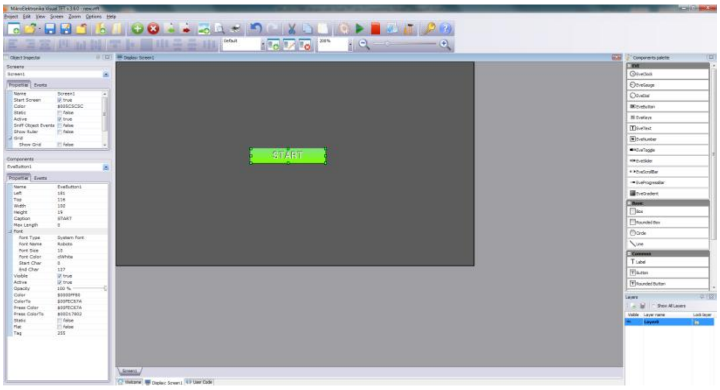
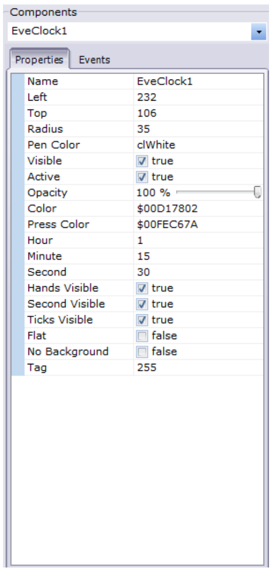


Figure 4.8 Objects Placement

Multiple screens are also possible by pressing the  icon.

This allows for a button press or a touch screen event to load a second screen and display new information.

4.4 Object Properties



The properties of an object such as colour, size and location may be configured in the component property table. This allows for configuration to happen without writing any code. For complex objects, such as the clock widget, additional settings such as the time may be set via this property table also.

Figure 4.9 Object Properties

4.5 Generating Code

After placing the objects on the main window the next step is to generate code that the compiler will use to generate the executable binary. With Visual TFT this is achieved with a simple button press from the ribbon bar.

The  icon will generate the code and allow you to view the generated code.

At this stage the project contains enough information to build an output file that will configure the microcontroller SPI port and draw a static display. To make use of the touch features or create animations will require the user to write code.

4.6 User Code

User code allows buttons to perform functions or clocks to tick.

The first step is to create the event in the component property window.

This is the label for the actual event.

In the user code window to make the button switch from the current screen to screen2, the code would look like:

```
void EveButton1OnClick() {
   {DrawScreen(&Screen2);}
```

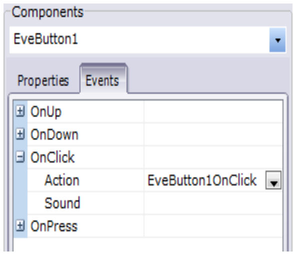


Figure 4.10 Object Properties

For tasks such as making the clock tick, code such as:

```
void Clock() {
    Delay_ms(1000);
    if (EveClock1.sec <= 59)
      {EveClock1.sec = (EveClock1.sec + 1);
       DrawScreen(&Screen2); }
    else {EveClock1.sec = 0;
          DrawScreen(&Screen2);
          if (EveClock1.min <= 59)
                {EveClock1.min = (EveClock1.min + 1);
                DrawScreen(&Screen2); }
          else {EveClock1.min = (EveClock1.min = 0);
                DrawScreen(&Screen2);
                if (EveClock1.hour <= 11)
                      {EveClock1.hour = (EveClock1.hour + 1);
                      DrawScreen(&Screen2); }
                else {EveClock1.hour = (EveClock1.hour = 0);
                      DrawScreen(&Screen2); }
                }
```

}

} is used.

Note the format for selecting the clock parameters:

EveClock1 is the object.

.min is the minute parameter

The value for a minute is between 0 and 59.

More information on MikroElectonica syntax is available from MikroElectronika or via the tool online help.

This code is then sent with the full project to the compiler.

To send the code to the compiler use the  icon in the ribbon toolbar.

5 Code Compiler

The files from Visual TFT are sent to the MikroElectronika compiler. This will auto generate a project and pull in the code created by visual TFT.

This code will all be based around MikroElectronika libraries. However the "events_code" file will match the text input user code in Visual TFT. This code may be edited in the compiler or the visual TFT tool as the projects are now interlinked.

From the Build option in the toolbar select "Rebuild all sources" to generate the HEX file that will be loaded into the processor.

The tool also contains a messages window such that any errors in the code may be found. These errors should hopefully be confined to the events_code file and not the auto generated sections.

6 Code Loading onto the Microcontroller

Loading the compiled file from the PC to the target microcontroller may depend on the microcontroller chosen. Some devices allow for programming over UART, USB or JTAG.

At FTDI we use the mikroProg tool from MikroElectronika. This is integrated into the MikroElectronika tool suite and allows for rapid programming and debug.

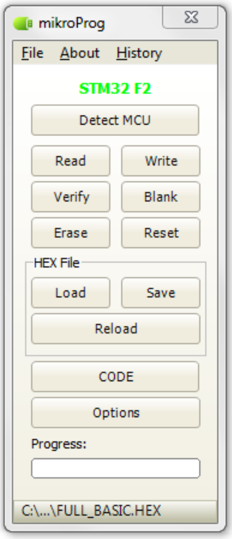


Figure 6.1 mikroProg

7 FT800 Design Summary

The FT800 provides an easy way to incorporate graphics displays into products providing for a lower cost solution or enabling a display into systems that could not otherwise afford this capability. With only the FT800 between the MCU and the LCD display, a vivid graphics experience with touch and audio is now possible.

The overall design flow regardless of the toolset chosen, from component selection to displaying the first screen is captured here:

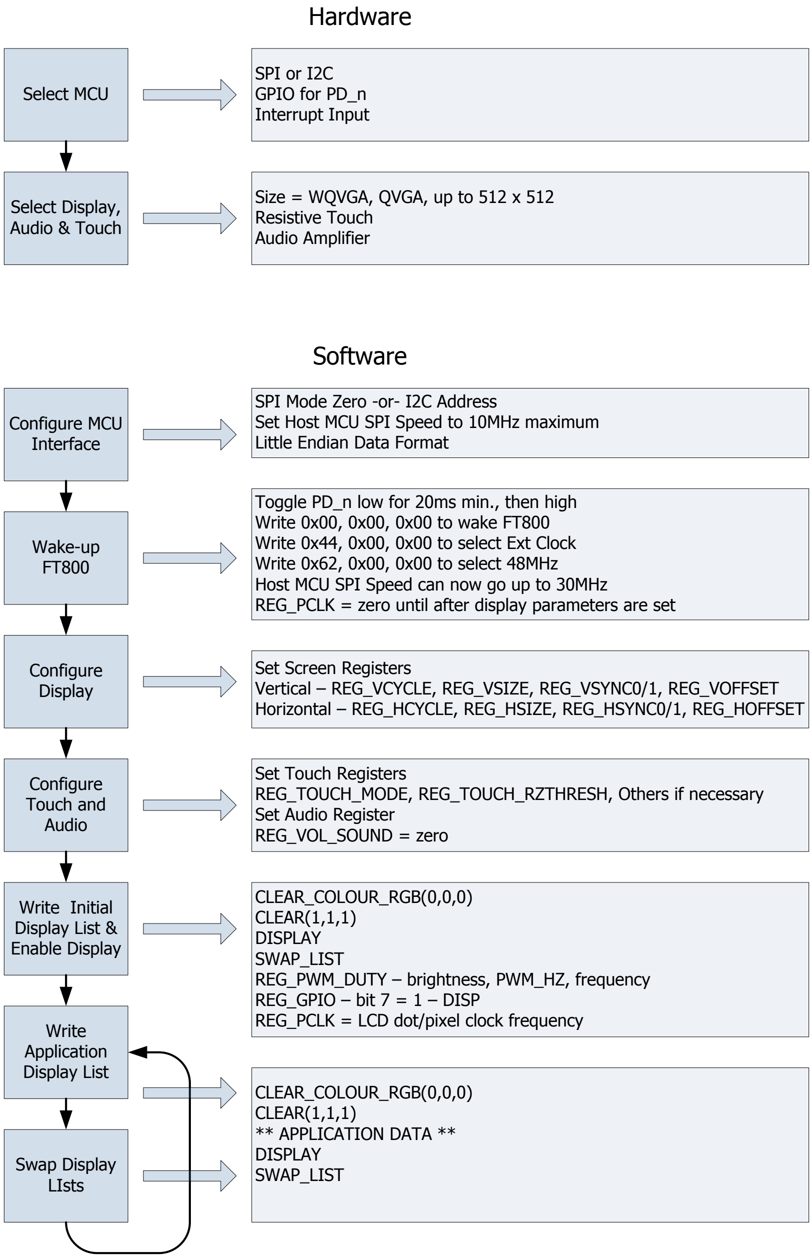


Figure 7.1 FT800 Hardware and Software Design Flow

8 MikroElektronika

MikroElektronika have been developing world class tool chains for microcontroller application development for over a decade. To find out more about MikroElektronika and more importantly the FT800 support they provide, visit their website at www.miikroe.com

9 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

MikroElectronika (www.mikroe.com)

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)

Appendix B – List of Tables & Figures

List of Figures

Appendix C – Revision History

Document Title: AN_254 FT800 Designs With Visual TFT

Document Reference No.: FT_000880

Clearance No.: FTDI# 350

Product Page: http://www.ftdichip.com/EVE.htm

Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2013-09-30