



Application Note

AN_277

FT800 Custom User Font Implementation

Version 1.1

Issue Date: 2015-09-29

This application note describes how to create user-defined fonts and utilize them with the FT8xx series of devices. This allows different font styles, languages and special characters to be used within the application.

Use of FTDI devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold FTDI harmless from any and all damages, claims, suits or expense resulting from such use.

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>

Copyright © 2015 Future Technology Devices International Limited

Table of Contents

1	Introduction	2
1.1	Scope	2
1.2	Software Required.....	2
1.3	Hardware Required	2
1.4	Limitations of Fnt_cvt.exe	2
2	Introducing the Font converter utility	3
2.1	Output files	3
2.2	Commands	3
3	Operational flow of font conversion.....	5
3.1	Make a source text file	5
3.2	Converting font by fnt_cvt.exe	7
4	Example Code	9
4.1	Example of displaying a Custom Font	9
4.2	An example of setfont2	11
5	Conclusion	12
6	Contact Information	13
Appendix A – References		14
	Document References	14
	Acronyms and Abbreviations.....	14
Appendix B – List of Tables & Figures		15
	List of Tables.....	15
	List of Figures	15
Appendix C – Revision History		16

1 Introduction

This application note describes how to create and utilize user-defined fonts with the FT8xx Series. The raw font data is loaded into the graphics RAM of the FT8xx by the host and can then be used via the FT8xx's internal font table in a similar manner to the built-in fonts.

1.1 Scope

This document shows how to generate raw data from a user-defined font with FTDI's font converter utility <http://www.ftdichip.com/Support/Utilities.htm#EVEFontConverter> on a Windows PC, and shows how to add the raw data to the font table of the FT8xx. Example code is also provided to explain how to output the user-defined characters on a display.

1.2 Software Required

- Microsoft Visual Studio with Visual C++ 2010, which can be downloaded from the Microsoft website.
<https://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx>
- Fnt_cvt.exe EVE font converter utility provided by FTDI. It can be downloaded from following link.
<http://www.ftdichip.com/Support/Utilities.htm#EVEFontConverter>
- FTDI D2XX driver, which can be downloaded from FTDI's website.
<http://www.ftdichip.com/Drivers/D2XX.htm>

1.3 Hardware Required

- PC with Windows 7 OS or later installed.
- FT8xx Series development module with LCD panel e.g. VM800B, VM800C, VM801B or VM810C50A-D.
- FTDI MPSSE adapter (e.g. C232HM-DDHSL-0 cable or VA800A-SPI module)

1.4 Limitations of Fnt_cvt.exe

- Only non-cursive fonts are supported
- Bitmap handle 0 to 14 can be used for user-defined fonts
- Bitmap handle 15 is a scratch bitmap handle for CMD_GRADIENT, CMD_BUTTON and CMD_KEYS commands. If the aforementioned commands are not being used in the Display List then the bitmap handle can also be used for the user-defined fonts. NOTE: FT81x has the ability to redefine the scratch bitmap handle to a different bitmap handle but the default is still 15
- A maximum of 127 characters can be saved in a font table, index 0 is reserved.
- A source text file must be saved in UTF-8 format.

2 Introducing the Font converter utility

Fnt_cvt.exe is the utility which runs on Windows and extracts characters from the font file into a specified point size, FT8xx Series metric block, and raw bitmap data. The characters to be converted must be specified in a UTF-8 encoding file or ASCII printable code set from 32 to 126.

2.1 Output files

The EVE font conversion utility will generate the metric block file as well as L1, L2 (FT81x only), L4, L8 format bitmap data. The output is a 148byte metric block appended with the variable size raw bitmap data.

The output data of this utility is prepared for 1 bitmap handle of the FT8xx Series.

The EVE font conversion utility will generate three files as follows in each of the L1, L2 (FT81x only), L4 and L8 folders.

- *.raw: The binary format of the converted file, which can be downloaded into FT8xx graphics memory directly with the BITMAPS command.
- *.rawh: The header file of the converted file, which is in text representation. The programmer can include this file into their program and build it into the final binary.
- *.rtf: The file defining the character and index mapping relationship.

The following table shows the format of font metric block

Address	Size	Value
p + 0	128	width of each font character, in pixels
p + 128	4	font bitmap format, for example L1, L2(FT81x only), L4 or L8
p + 132	4	font line stride, in bytes
p + 136	4	font width, in pixels
p + 140	4	font height, in pixels
p + 144	4	pointer to font graphic data in memory

Table 2.1 Format of font metric block

2.2 Commands

The command format of the conversion utility is:

```
fnt_cvt.exe -i TrueTypeFontFile -s pointsize [-u utf8_file][[-a]][[-c Optional_argument] [-d FT8xx_address]
```

Arguments list:

'-i' : Mandatory argument. The input file shall be a true type font file(*.ttf,*.ttc) located in the same folder as the utility, otherwise fnt_cvt.exe will search the windows system font directory for the font file.

'-s' : Optional argument. The width in pixels of font characters to be converted. If this argument is not present, the size is assumed to be 12.

'-u' : Mandatory argument if the input file is encoding in UTF-8. The utf8_file specifies the file name. Fnt_cvt.exe will read this file and convert the characters in this file. Users can copy the characters to be converted into this file and ensure the file encoded with UTF-8. The numbers of characters shall be no more than 127.

'-a': Mandatory argument if users want to convert the ASCII printable character set, i.e. from 32 to 126. There is no input file required.

'-d': Optional argument. This argument defines the start address of the metric block in the FT8xx Series RAM_G. If it is not present, fnt_cvt.exe will assume RAM_G is the starting address. Because the bitmap raw data follows the metric block, the raw bitmap data address is starting from FT8xx_address + 148.

'-c': Optional argument. Valid options are:

- | | |
|----------|---|
| setfont | Generate FT80x compatible font metric table, the default option. |
| setfont2 | Generate FT81x compatible font metric table for Cmd_Setfont2 command. A subset of the printable ASCII characters can be specified in the input file for -u or -t arguments. NOTE: the generated files with this option are not compatible with FT80x. |

3 Operational flow of font conversion

The operation steps are shown in this chapter, Windows 7 is used for this example.

3.1 Make a source text file

To make a source text file, it is not enough to simply key in characters in a text editor. The font type should be selected from the font table to make sure those characters show on the LCD as expected.

- a. Select Control Panel -> All Control Panel items -> Fonts window in PC, and all the fonts installed are showed in the window.

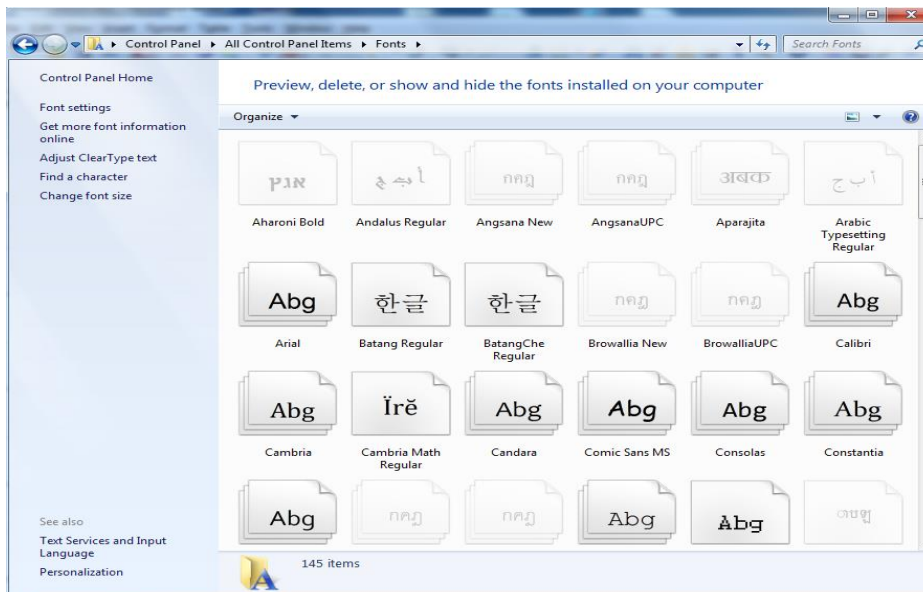


Figure 3-1 Font window

- b. Click "Find a character" in the left menu to pop up the Character Map window.
- c. Select a Chinese font from its drop down box. Here the FangSong font is selected as an example for Chinese font display.

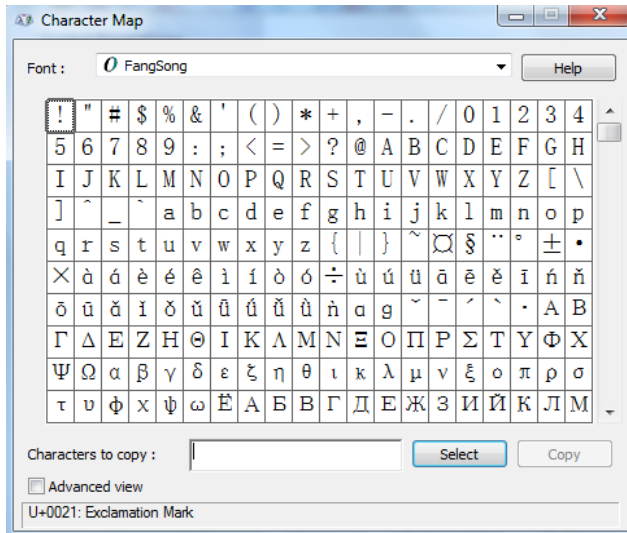


Figure 3-2 Character Map

- d. Select characters in this window. Here the standard and extended ASCII characters, and some Chinese characters are selected as an example.
- e. Select character "%" and click button "Select", then repeat this action for the next couple characters, or type the characters in the "characters to copy" box directly. The whole string in this example is "%9Mau é ð H γ ψ 中文字與特殊符號顯示範例".

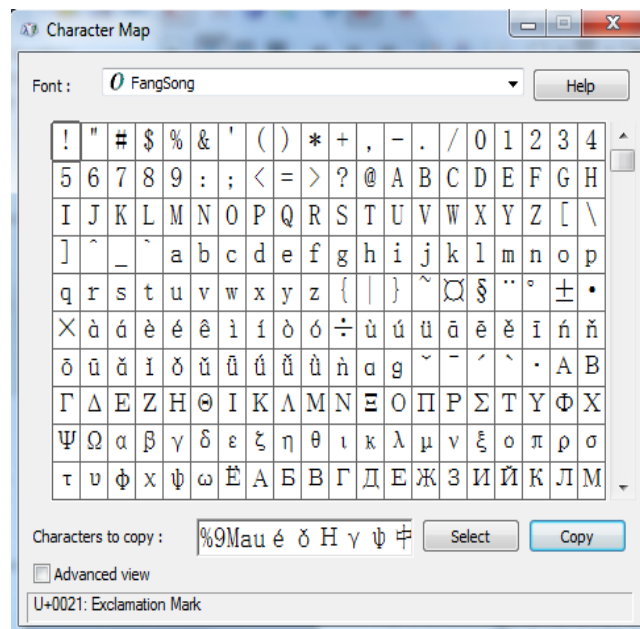


Figure 3-3 Character Map

- f. Click "Copy" button to finish it.
- g. open a text editor such as NotePad, and paste those characters into it.

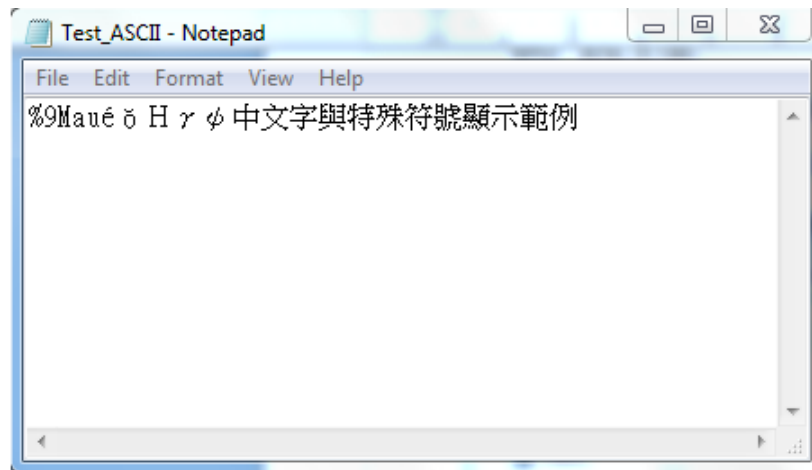


Figure 3-4 Text editor Notepad

- h. Click File -> Save, to save the file as a .txt file, and select encoding as UTF-8. Test_ASCII.txt is the name of this example text file.

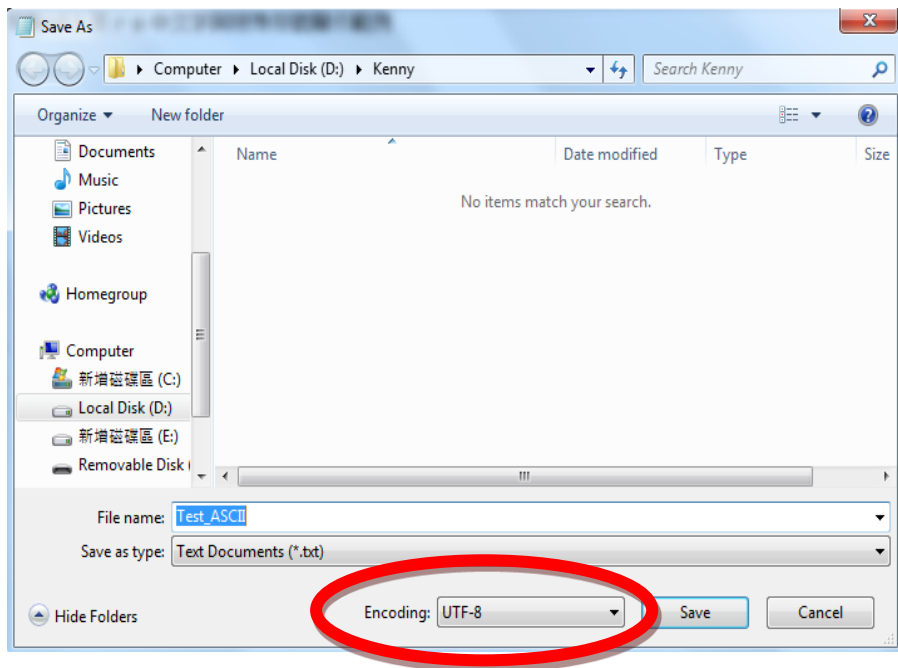
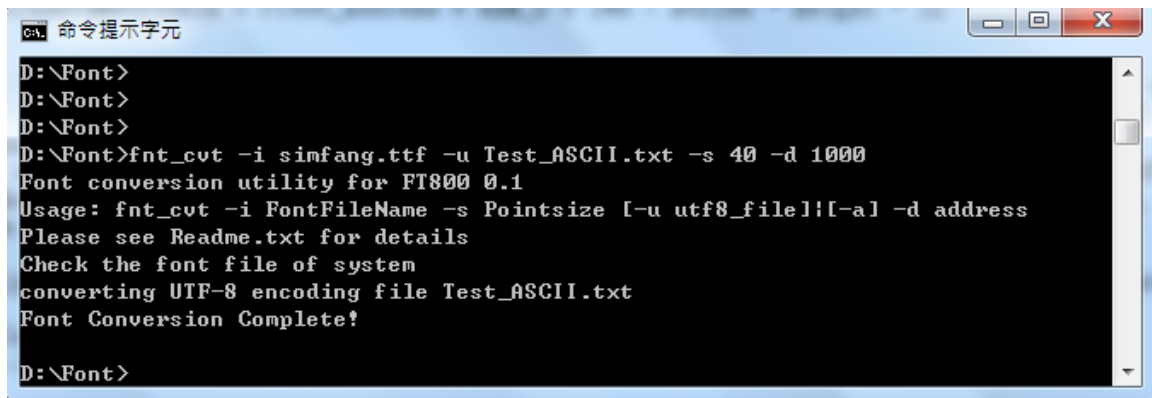


Figure 3-5 Window of Save text file

3.2 Converting font by fnt_cvt.exe

The text file should be converted into a raw or rawh file for loading into the RAM font table of the FT8xx for display. The operational flow is shown in this chapter.

- a. Execute cmd.exe to open a DOS command window
- b. Enter the folder which contains EVE’s font utility, it is D:\Font> in this example.
- c. Type the command `fnt_cvt -i simfang.ttf -u Test_ASCII.txt -s 40 -d 1000`
 The usage of parameters for this command can be found in chapter 2.2. simfang.ttf is the file name of font Fangsong, pixel size 40 and the start address of font is 1000 in this example.



```
C:\> 命令提示字元
D:\Font>
D:\Font>
D:\Font>
D:\Font>fnt_cvt -i simfang.ttf -u Test_ASCII.txt -s 40 -d 1000
Font conversion utility for FT800 0.1
Usage: fnt_cvt -i FontFileName -s Pointsize [-u utf8_file]![-a] -d address
Please see Readme.txt for details
Check the font file of system
converting UTF-8 encoding file Test_ASCII.txt
Font Conversion Complete!
D:\Font>
```

Figure 3-6 Execute fnt_cvt command

- d. If successful, a folder named "simfang_Test_ASCII.txt40" is created, and there are 4 sub folders L1, L2 (FT81x only), L4 and L8 inside, which contain 4 kinds of raw bitmap data. Those raw format files are now ready for use with the FT8xx Series.


```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,151,255,98,0,0,0,82,255,166,0,0,0,193,129,0,0,  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,168,255,91,0,0,0,75,255,183,0,0,43,  
:  
:  
Skip.....  
};
```

c. Create sample code to show custom font

```
ft_void_t SAMAPP_ShowCustomFont()  
{  
    ft_prog_uchar8_t *special;  
    unsigned char i=0;  
    //load font data into memory  
    Ft_Gpu_Hal_WrMemFromFlash(phost, RAM_G + 1000, SAMApp_ShowCustomFont_MetricBlock, 148);  
    Ft_Gpu_Hal_WrMemFromFlash(phost, RAM_G + 1000 + 148, SAMApp_ShowCustomFont_FontBmpData,  
36432);  
    //start command  
    Ft_Gpu_CoCmd_Dlstart(phost);  
    //set the background to white color  
    Ft_App_WrCoCmd_Buffer(phost,CLEAR_COLOR_RGB(255,255,255));  
    Ft_App_WrCoCmd_Buffer(phost,CLEAR(1,1,1));  
  
    Ft_App_WrCoCmd_Buffer(phost,BEGIN(BITMAPS));  
    Ft_App_WrCoCmd_Buffer(phost,COLOR_RGB(255,0,0));//configure text as red color  
    Ft_App_WrCoCmd_Buffer(phost,BITMAP_HANDLE(6));//assign bitmap handle 6 as custom font  
table  
    //parameter value of below commands can be found in SAMApp_ShowCustomFont_MetricBlock[]  
    Ft_App_WrCoCmd_Buffer(phost,BITMAP_SOURCE(-508));// 1000 - (stride*height) + 148  
  
    Ft_App_WrCoCmd_Buffer(phost,BITMAP_LAYOUT(L8,36,46));  
    Ft_App_WrCoCmd_Buffer(phost,BITMAP_SIZE(NEAREST,BORDER,BORDER,36,46));  
    Ft_Gpu_CoCmd_SetFont(phost, 6, RAM_G +1000);  
    //put custome fonts to the expect X,Y coordinate  
    //we show those fonts in two rows  
    for(i=1;i<=10;i++)  
    {  
        //show font 1~10 in low 1  
        Ft_App_WrCoCmd_Buffer(phost,VERTEX2II(i*36,20,6,i));  
        //show font 11~20 in low 2  
        Ft_App_WrCoCmd_Buffer(phost,VERTEX2II(i*36,56,6,i+10));  
    }  
  
    Ft_App_WrCoCmd_Buffer(phost,END());  
    Ft_App_WrCoCmd_Buffer(phost,DISPLAY());  
    Ft_Gpu_CoCmd_Swap(phost);  
  
    /* Download the commands into fifo */  
    Ft_App_Flush_Co_Buffer(phost);  
  
    /* Wait till coprocessor completes the operation */  
    Ft_Gpu_Hal_WaitCmdfifo_empty(phost);  
}
```

4.2 An example of setfont2

This example shows the use of the SETFONT2 command.

```
//simpler method to load RAM font. Use the font conversion utility to convert the desired
subset of the ASCII characters, load font data, and use cmd_setfont2 command.
ft_void_t SAMAPP_CoPro_Setfont2() {
ft_uint16_t fontFileAddress = 100; //the converted font file will have a starting address of
RAM_G or 0th byte mark
Ft_Gpu_CoCmd_Dlstart(phost);
Ft_App_WrCoCmd_Buffer(phost, CLEAR(1, 1, 1));
Ft_App_WrCoCmd_Buffer(phost, COLOR_RGB(255, 255, 255));
SAMAPP_LoadRawFromFile("../..\\..\\Test\\Tuffy.ttf_24_L2.raw", fontFileAddress); //load font
data consists of font metric block and font data from the FTDI font conversion utility. The
file has been packaged with the font metric block at the beginning of the font data.
// SAMAPP_LoadRawFromFile("../..\\..\\Test\\simfang.raw", fontFileAddress); //load font data
consists of font metric block and font data from the FTDI font conversion utility. The file has
been packaged with the font metric block at the beginning of the font data.

Ft_App_WrCoCmd_Buffer(phost, BITMAP_HANDLE(1)); //associate to font handle 1
Ft_App_WrCoCmd_Buffer(phost, BITMAP_SOURCE(fontFileAddress + 148)); //The starting address for
the font file 148 bytes into the font file because the first 148 bytes are font metric
informational block.

Ft_App_WrCoCmd_Buffer(phost, BITMAP_LAYOUT(L2, 5, 29)); //font stride and height metrics can be
found in the .rawh file
Ft_App_WrCoCmd_Buffer(phost, BITMAP_SIZE(NEAREST, BORDER, BORDER, 20, 29)); //font width and
height metrics can be found in the .rawh file

Ft_Gpu_CoCmd_SetFont2(phost, 1, fontFileAddress, 97); //set ram font to font handle 1, font
metric block starts at fontFileAddress(RAM_G), and set the starting character to the 32th
character in the set.

Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)-30, (FT_DispHeight / 2) - 80, 1, OPT_CENTER, "f");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)-15, (FT_DispHeight / 2) - 80, 1, OPT_CENTER, "t");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)+15, (FT_DispHeight / 2) - 80, 1, OPT_CENTER, "d");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)+30, (FT_DispHeight / 2) - 80, 1, OPT_CENTER, "i");
#ifdef ARDUINO_PLATFORM || defined(FT900_PLATFORM)
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)-30, (FT_DispHeight / 2) - 80, 29, OPT_CENTER, "f");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)-15, (FT_DispHeight / 2) - 80, 29, OPT_CENTER, "t");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)+15, (FT_DispHeight / 2) - 80, 29, OPT_CENTER, "d");
Ft_Gpu_CoCmd_Text(phost, (FT_DispWidth / 2)+30, (FT_DispHeight / 2) - 80, 29, OPT_CENTER, "i");
#endif

Ft_App_WrCoCmd_Buffer(phost, DISPLAY());
Ft_Gpu_CoCmd_Swap(phost);
Ft_App_Flush_Co_Buffer(phost);
Ft_Gpu_Hal_WaitCmdfifo_empty(phost);
SAMAPP_ENABLE_DELAY_VALUE(5000);
}
```

5 Conclusion

The test result of the above sample code is shown in the following figure. It is now possible to add and utilize these characters in a display, which could be needed for applications in specific regions or markets.

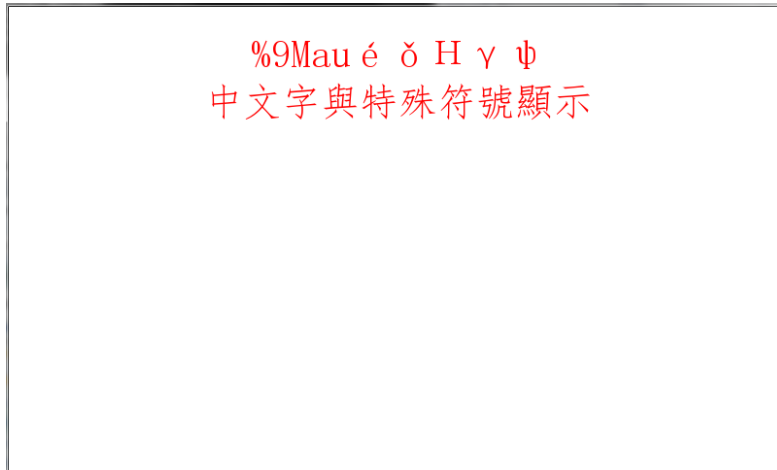


Figure 5-1 Test result of 4-1 sample code



Figure 5-2 Test result of 4-2 sample code

6 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

<http://ftdichip.com>

Distributor and Sales Representatives

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

System and equipment manufacturers and designers are responsible to ensure that their systems, and any Future Technology Devices International Ltd (FTDI) devices incorporated in their systems, meet all applicable safety, regulatory and system-level performance requirements. All application-related information in this document (including application descriptions, suggested FTDI devices and other materials) is provided for reference only. While FTDI has taken care to assure it is accurate, this information is subject to customer confirmation, and FTDI disclaims all liability for system designs and for any applications assistance provided by FTDI. Use of FTDI devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold harmless FTDI from any and all damages, claims, suits or expense resulting from such use. This document is subject to change without notice. No freedom to use patents or other intellectual property rights is implied by the publication of this document. Neither the whole nor any part of the information contained in, or the product described in this document, may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. Future Technology Devices International Ltd, Unit 1, 2 Seaward Place, Centurion Business Park, Glasgow G41 1HH, United Kingdom. Scotland Registered Company Number: SC136640

Appendix A – References

Document References

N/A

Acronyms and Abbreviations

Terms	Description
EVE	FTDI's Embedded Video Engine
LCD	Liquid-Crystal Display, is one type of display device
USB	Universal Serial Bus, a popular I/O interface for electronic peripherals

Appendix B – List of Tables & Figures

List of Tables

Table 2.1 Format of font metric block	3
---	---

List of Figures

Figure 3-1 Font window	5
Figure 3-2 Character Map.....	6
Figure 3-3 Character Map.....	6
Figure 3-4 Text editor Notepad	7
Figure 3-5 Window of Save text file.....	7
Figure 3-6 Execute fnt_cvt command	8
Figure 5-1 Test result of 4-1 sample code	12
Figure 5-2 Test result of 4-2 sample code	12

Appendix C – Revision History

Document Title: AN_277 FT800 Custom User Font Implementation
Document Reference No.: FT_000939
Clearance No.: FTDI# 368
Product Page: <http://www.ftdichip.com/FTProducts.htm>
Document Feedback: [Send Feedback](#)

Revision	Changes	Date
1.0	Initial Release	2013-11-25
1.1	Add L2 format and setfont for FT81x	2015-09-29