UEFI Development Exploration 15 – Text Display in Graphics Mode

原创 luobing4365 Posted on 2019-09-13 20:43:42 Read 1.9k Collection 2 Likes 2

Category Column: UEFI Development Article Tags: UEFI Programming UEFI Graphics Programming Low-level programming Small font library UEFI Chinese characters display

(Please keep it-> Author: Luo Bing https://blog.csdn.net/luobing4365)

UEFI provides **HII (Human Interface Infrastructure)** to handle text display in the graphical interface, but I have not studied this carefully. So far, I have not used this method to display Chinese characters and text.

For example, in the previous blog, I demonstrated the interface of the test sample card. Various English and Chinese characters were used in the graphics, all of which were implemented using my own methods.

HII's way of thinking was not very direct. At that time, I was faced with the task of completing the development of the UEFI Option ROM within a few weeks. There were too many difficulties to overcome. For the display of Chinese and English characters in graphic mode, I adopted the display method in Foxdisk.

Specifically, in the previous blog, we have implemented the putpixel function, so we can treat all Chinese and English characters as fonts and draw them. All the core ideas come from this.

In the blog of Foxdisk programming, some content of this part has been introduced, mainly about the extraction and display of Chinese characters, which can be compared. Because HII is not used, the compiler cannot recognize the unicode characters in the source code. Therefore, all the Chinese character strings I use are like this:

```
UINT8 str_sywai[]={0x31,0x2e,0x20,0xbd,0xf8,0xc8,0xeb,0xcd,0xe2,0xcd,0xf8,0x00};
UINT8 str_synei[]={0x32,0x2e,0x20,0xbd,0xf8,0xc8,0xeb,0xc4,0xda,0xcd,0xf8,0x00};
UINT8 STR_CNT_WAI[] = {0x31,0x35,0x20,0xc3,0xeb,0xba,0xf3,0xbd,0xf8,0xc8,0xeb,0xb5,0xbl,0xc7,
UINT8 STR_CNT_NEI[] = {0x31,0x35,0x20,0xc3,0xeb,0xba,0xf3,0xbd,0xf8,0xc8,0xeb,0xb5,0xbl,0xc7,
```

Figure 1 String (from Mainmenu.c)

That is, the Chinese character encoding is directly represented by two 8-bit characters (the Chinese character encoding is 16 bits wide), and the English character is the ASCII code (8 bits). The tool program automatically extracts the font according to the encoding.

The tool programs under Foxdisk also need to make some simple changes to adapt to the current program requirements.

1. Tools

There is no need to extract the font of English characters. I have always used 8×16 pixel English characters. Just copy them into the program and compile them together.

The tool is mainly used to extract Chinese character fonts. The tool I wrote runs under 16-bit DOS, so it can only be run in the cmd of 32-bit WinXP. Win7 32-bit should also work. The tool program mainly processes file contents and does not call hardware- related codes. It is compiled using Borland C++ 3.1.

Forget about using a 64-bit operating system. I wanted to use Visual Studio to write code to adapt to a 64-bit OS, but it was not necessary in my daily work, so I was always unmotivated and kept putting it off until now.

I'm thinking about rewriting these tools in Python to run them directly on x64. Maybe I'll do it, maybe I won't, who knows.

There are two main tool programs used: DistillHZ and DisTillLOGO. Both programs are modified from Foxdisk's tool program to meet the needs of UEFI development. DistillHZ is modified to automatically generate UEFI strings (the strings I need, not the strings officially stated by UEFI). You can see it by looking at the program log:

```
/*exetrace hz form files*/
/*luobing 2008-10-20 12:29*/
/*程序功能：可提取最大32个文件中由" "包含的汉字，并提取出其字模*/
/*注意：    必须保证在提取的文件中字符串"*.*"中没有//和/* 字符,否则可能会提取错误*/
/*luobing 2011-09-06 21:28:11
  在UEFI的环境下显示汉字，目前没有找到方法以汉字形式来定义字符串，比如"我"在AMI的
  编译环境下是显示不出来的。因此增加一个缓冲机制，每个需要显示的字符串都定义为形
  如 UINT8 Str[]={0x12,0x34,...0x00};的样子，然后再通过相应的字符串显示函数显示*/
/*为了处理上的简单，源文件中的字符串都定义为UINT8 Str[]="吾爱吾妻forever";之类的形式*/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <dos.h>
#include <time.h>
#include "ver_l.h"

unsigned char getHZ(char *dstfile,char *srcfile);
unsigned char ConvertStr(char *srcfile);

char  szHelpContext[] = \
"Syntax is:  efihz.exe dst src1 src2 ...\n"
" /V        Version control     \n"

" /? or /H  Display Help file\n";
```

Figure 2 DisTillHZ program description

In short, compared to Foxdisk's original tool, in addition to automatically generating font files, it also automatically converts the required strings.

If the string to be converted is placed in t1.c, run: efihz xx.h t1.c, the Chinese character library will be automatically extracted to xx.h, and a dst_t1.c file will be automatically generated, which contains the string used by the uefi program.

DisTillLOGO is completely taken from the Foxdisk tool program without modification. The logo extraction is not perfect yet, and the size needs to be manually modified in the code, and the impact of various situations (such as the alignment of bmp) is not considered.

See the code and the instructions within the code for details.

2Display Chinese characters

In fact, Foxdisk's blog has clearly explained the principle of displaying Chinese characters. I use exactly the same mechanism, so I won't repeat it. The difference lies in the representation of the string of the UEFI program, which is also solved by the tool program mentioned above.

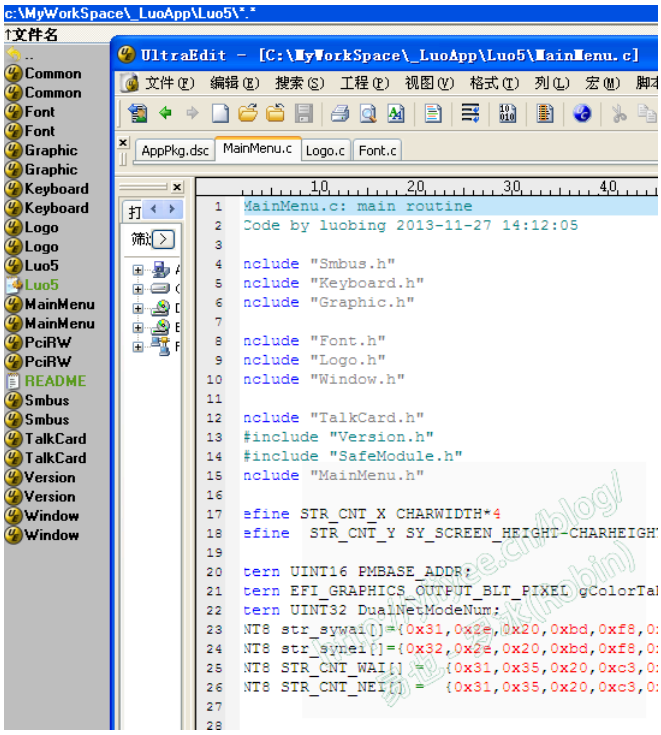

Figure 3 Code List

Use distillhz to extract the Chinese fonts and strings that need to be displayed, and copy the font files to Font.c. Font.h contains the data structure for displaying Chinese and English characters, as well as related display functions, which basically do not need to be changed.

The so-called Logo is actually what is needed to be displayed on the title bar of the program, similar to the logo of the program under the window. I extracted it from the 256-color bmp image and stored it in the form of pixels. When displaying it, I use putpixel to display it point by point, which is relatively simple.

A picture is worth a thousand words. The program displays the following effect:

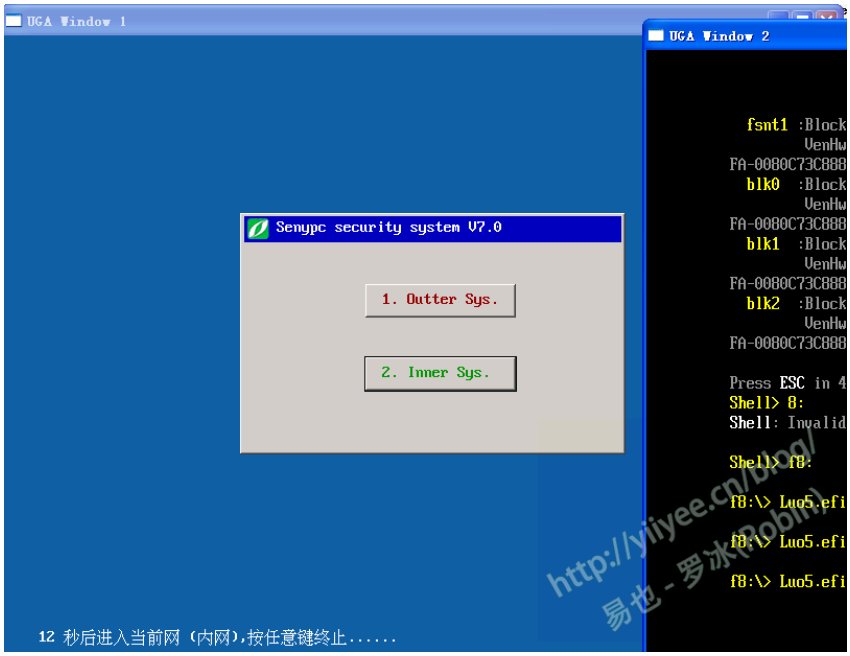

Figure 4 The display effect of the program

From the code list, we can see that many source code names are consistent with those in Foxdisk. In fact, the display functions of Chinese characters and various graphics are completely consistent with those in Foxdisk. Therefore, some interesting effects implemented in Foxdisk, such as fading effects and transparent effects, can be directly used in UEFI.

It is because of this kind of fun that I never get tired of this kind of research.

This blog was written at the 7 Days Inn next to the Fanshen subway station in Shenzhen. I borrowed the hotel's TV screen, which was a bit funny, so I took a picture of it as a souvenir.

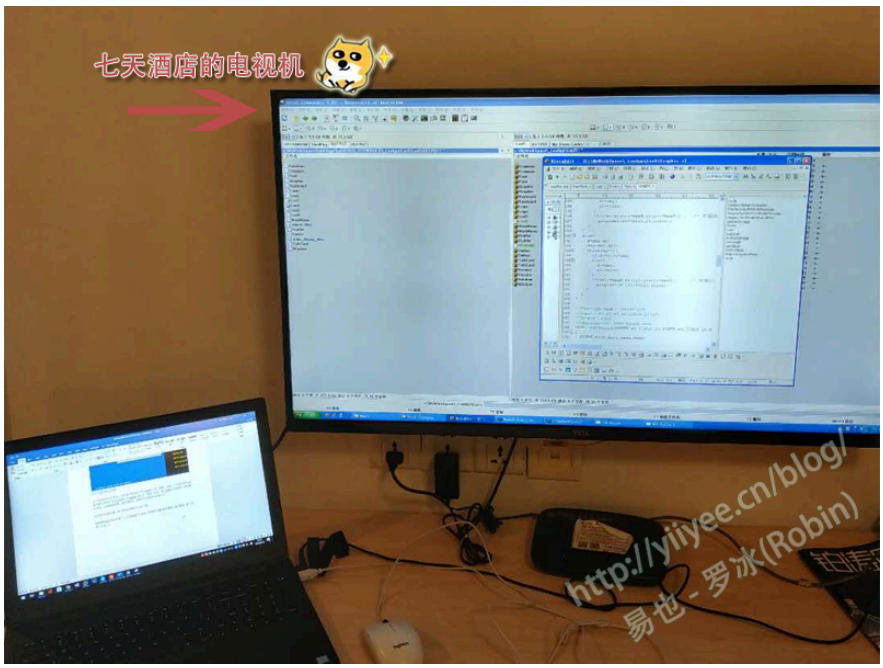

Figure 5: Leave a souvenir

I guess no one would use the hotel TV like me^_^

Gitee address: https://gitee.com/luobing4365/uefi-explorer
The project code is located under: / X3 UEFI interface development and tool programs.