

GLCD_FontCreator

Quickguide V 0.5

About ...

GLCD_FontCreator (Windows: .Net 4.5.2; VisualStudio 2015)

Create GLCD bitmap fonts from installed and independent TrueType fonts.

The architecture supports integration of different output formats. See FC_Template.cs

Currently implemented is: **GLCD_FC2_Compatible** Compatible output code format for AVR projects (Arduino ..)
using - GLCD Arduino library support for graphic LCDs by Michael Margolis and Bill Perry

Licensed under the Apache License, Version 2.0

Credits:

Original FontCreator: FontCreator2.0 resp. 2.1 from F. Maximilian Thiele

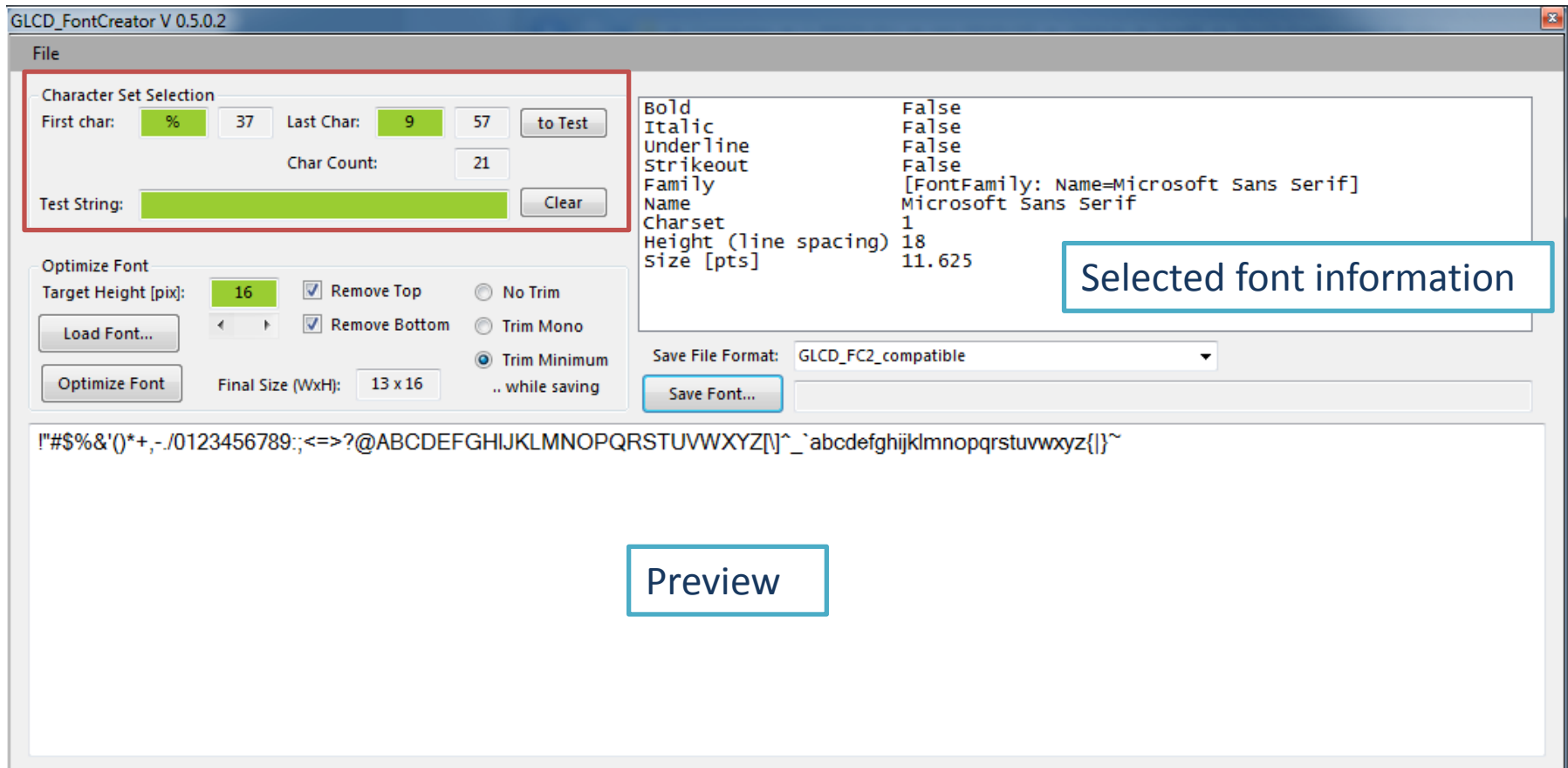
Font Dialog derived from CustomFontDialog by Syed Umar Anis (<http://umaranis.wordpress.com/>)

Note: Fonts are copyrighted materials. Before using any derived bitmap make sure you are compliant with the license.

There are a number of great free to use fonts out there.

But don't distribute/sell derived fonts if the license is not explicitly allowing you this.

The GUI – Character Set Section



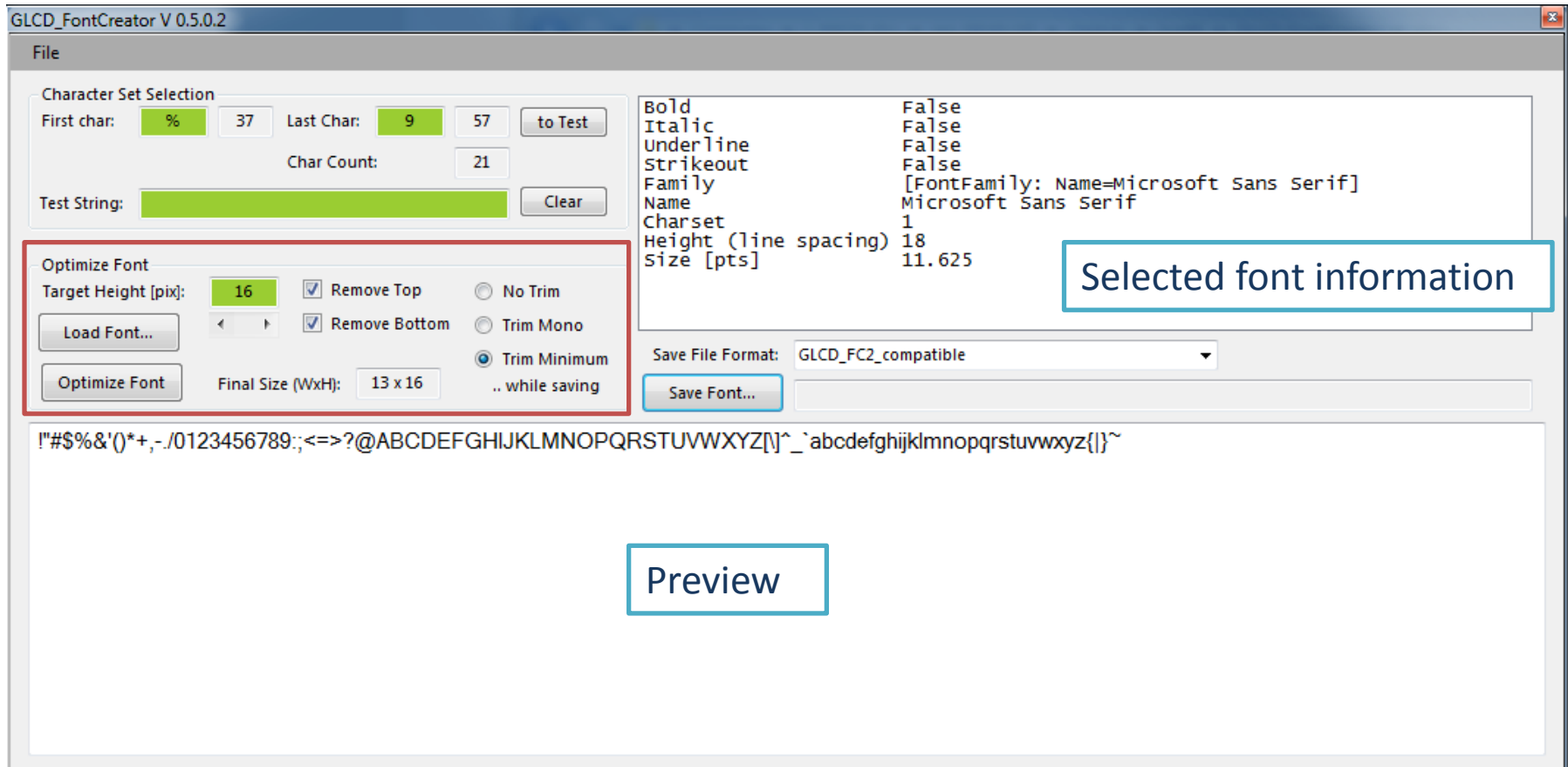
“First Char”: the letter where the font file will start with (type any character here – NOTE this is ASCII 7 bit only)

“Last Char”: the letter where the font file will end with (type a letter that is beyond the ordinal of the starting letter)
- Ordinals are shown after the letter (e.g. % -> 37)

Bt. “To Test”: copies the full sequence into the “Test String” field and will show this then in the preview part

Bt. “Clear”: clears “Test String” – Preview shows the printable ASCII 7-bit charset

The GUI – Optimize Font



A font needs to be optimized to extract the bitmap.

Bt. "Load Font...": Opens the font selection dialog to choose a font to process

Bt. "Optimize Font": Click and the font will be optimized to fit the targets set.

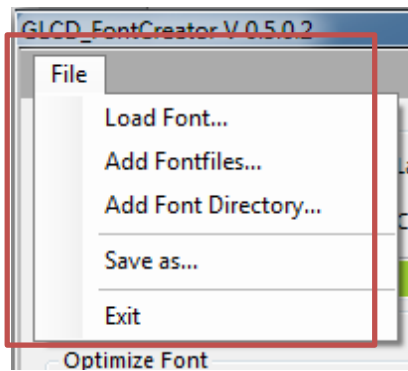
"Target Height": click the arrows to get a larger or smaller target height in pixels.

"Remove Top" – deletes blank pixels from top that are common to all characters in the selected set

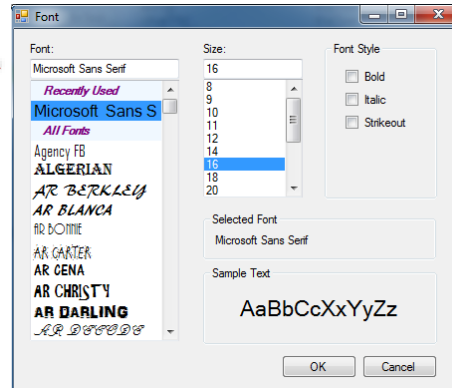
"Remove Bottom" – deletes blank pixels from the bottom that are common to all characters in the selected set

"No Trim", "Trim Mono", "Trim Minimum" – width trimming that is applied to the bitmap creation while saving the font

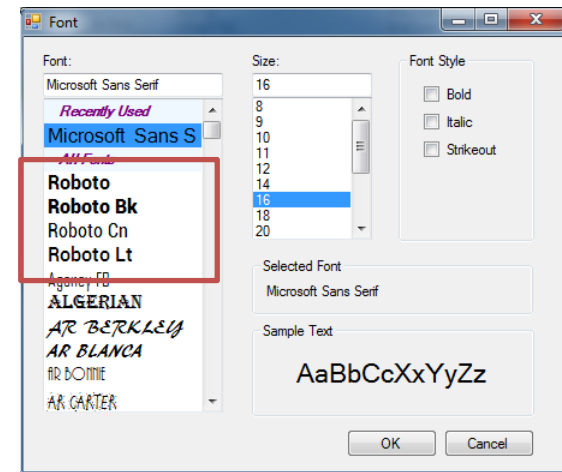
The GUI – The File Menu



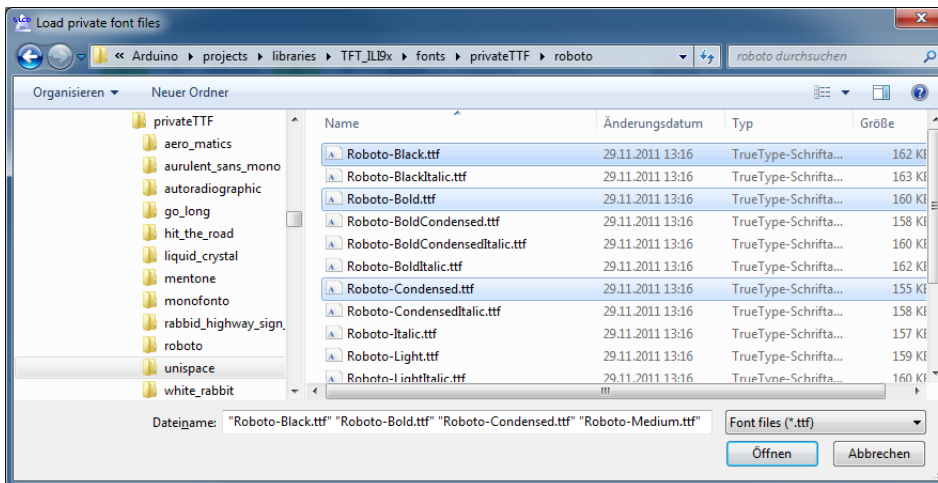
Add font files ...



Load a font ..
without added user fonts



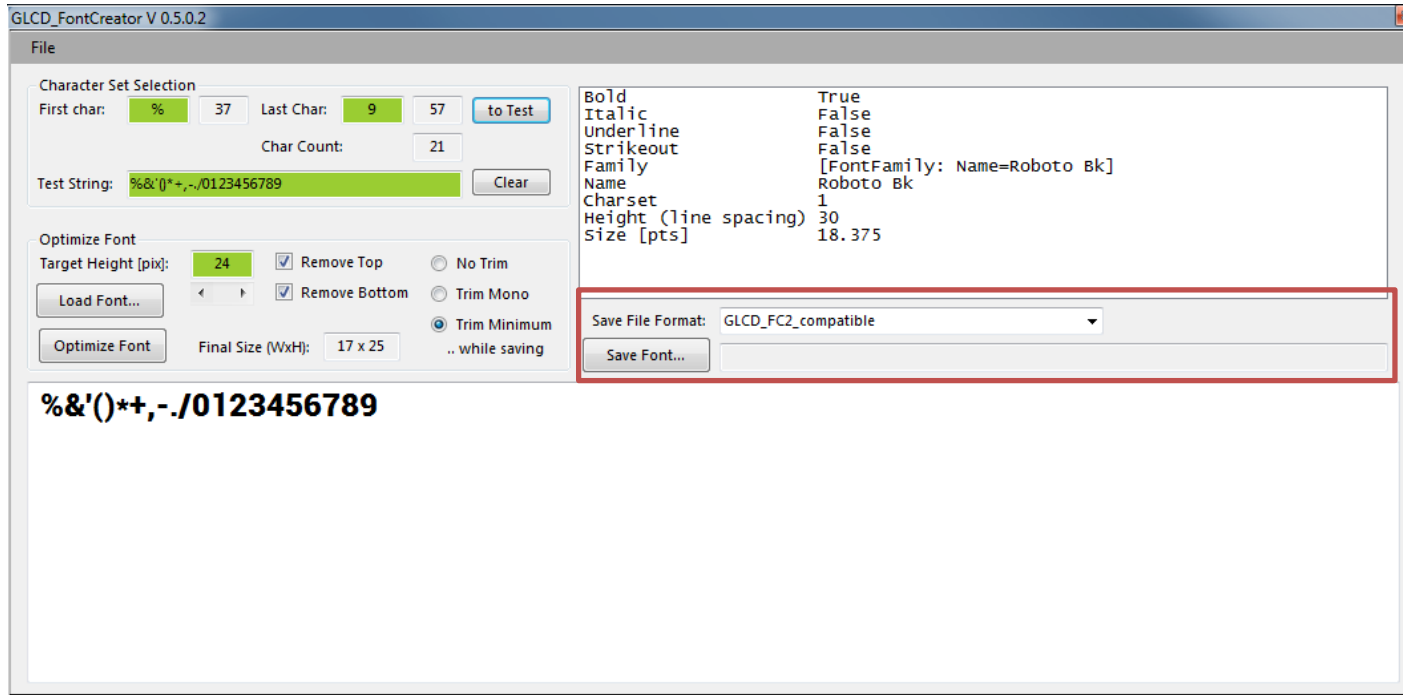
Load a font ..
after added user fonts



The program allows you to temporarily add font files that are not part of the installed font collection. Use either “Add Fontfiles...” to choose them or “Add Font Directory” to load all TTF files in a folder and its subfolders. Such added fonts are maintained at the top of the “All Fonts” part of the font chooser. The fonts are available until closing the program.

Note: Only valid TrueType fonts are supported (this is a limitation of the used WindowsForms GDI+ infrastructure)

The GUI – Save Font...



Select a Font processor with “Save File Format” (currently only GLCD_FC2_Compatible is available)
Hit “Save Font...” and direct it to the folder you want to save the created header file.

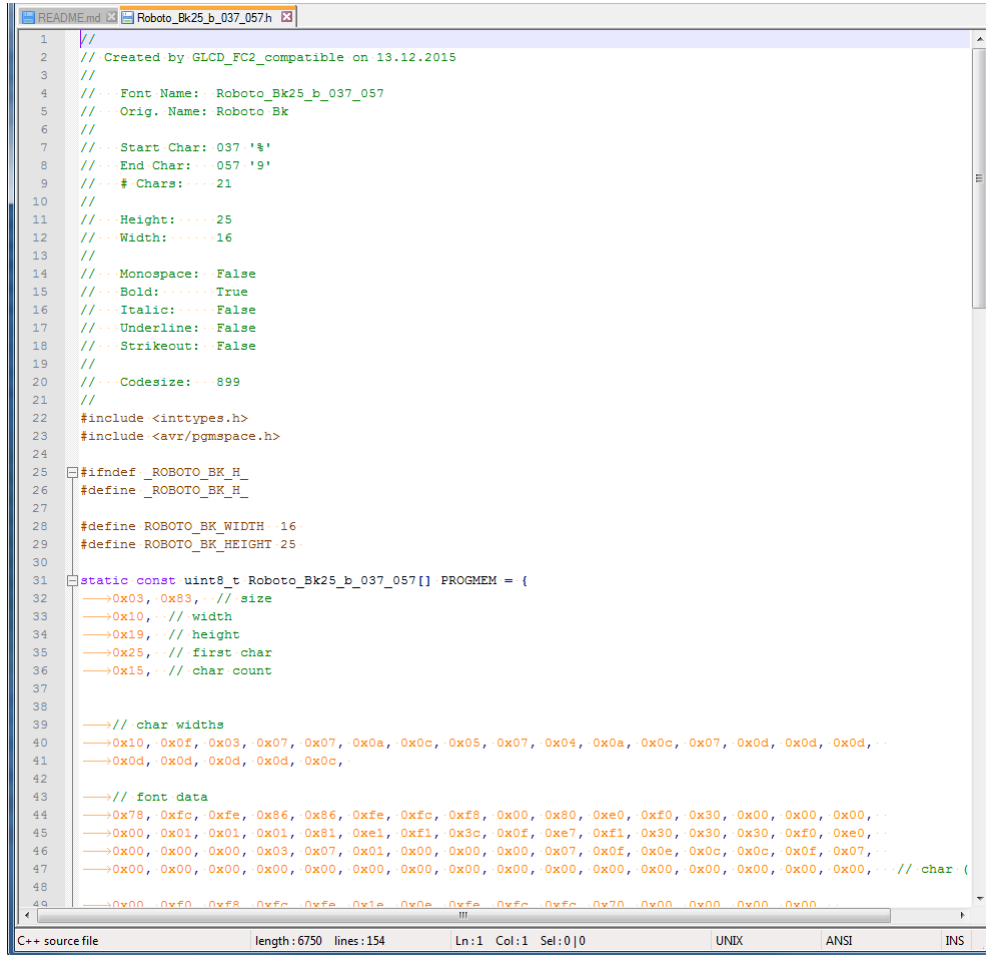
The example above would create a font map with numbers and signs as shown (“to Test” used) optimized for the Height of 24 pix.

Note: due to the available font scaling in Windows a particular font will sometimes not fit the exact target – it takes the next larger one then. In the example it can only achieve heights of 23 or 25 (this is NOT a bug...)

Saving the example will create 2 files: Roboto_Bk25_b_037_057.h and Roboto_Bk25_b_037_057.h.png (thumbnail of the created font)

%&'()*+,-./0123456789

The Output



```
1 //
2 // Created by GLCD_FC2_compatible on 13.12.2015
3 //
4 // Font Name: Roboto_Bk25_b_037_057
5 // Orig. Name: Roboto Bk
6 //
7 // Start Char: 037 '!'
8 // End Char: 057 '9'
9 // # Chars: 21
10 //
11 // Height: 25
12 // Width: 16
13 //
14 // Monospace: False
15 // Bold: True
16 // Italic: False
17 // Underline: False
18 // Strikeout: False
19 //
20 // Codesize: 899
21 //
22 #include <inttypes.h>
23 #include <avr/pgmspace.h>
24
25 #ifndef _ROBOTO_BK_H_
26 #define _ROBOTO_BK_H_
27
28 #define ROBOTO_BK_WIDTH 16
29 #define ROBOTO_BK_HEIGHT 25
30
31 static const uint8_t Roboto_Bk25_b_037_057[] PROGMEM = {
32     →0x03, 0x83, // size
33     →0x10, // width
34     →0x19, // height
35     →0x25, // first char
36     →0x15, // char count
37
38     →// char widths
39     →0x10, 0x0f, 0x03, 0x07, 0x07, 0x0a, 0x0c, 0x05, 0x07, 0x04, 0x0a, 0x0c, 0x07, 0x0d, 0x0d, 0x0d,
40     →0x0d, 0x0d, 0x0d, 0x0d, 0x0c,
41
42     →// font data
43     →0x78, 0xfc, 0xfe, 0x86, 0x86, 0xfe, 0xfc, 0xf8, 0x00, 0x80, 0xe0, 0xf0, 0x30, 0x00, 0x00, 0x00,
44     →0x00, 0x01, 0x01, 0x01, 0x81, 0xf1, 0x3c, 0x0f, 0xe7, 0xf1, 0x30, 0x30, 0x30, 0xf0, 0xe0,
45     →0x00, 0x00, 0x00, 0x03, 0x07, 0x01, 0x00, 0x00, 0x00, 0x07, 0x0f, 0x0e, 0x0c, 0x0c, 0x0f, 0x07,
46     →0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
47     →0x00, 0xf0, 0xf8, 0xfc, 0xfe, 0xf1, 0xf5, 0xfc, 0xf8, 0x70, 0x00, 0x00, 0x00, 0x00,
48
49 }
```

C++ source file | length: 6750 | lines: 154 | Ln:1 Col:1 Sel:0|0 | UNIX | ANSI | INS

A pretty printed .h file containing bitmap data which can be used for AVR projects.