**FONT HEX FILE GENERATION STEP by STEP PROCEDURE**

**THREE STAGE OF PROCEDURE**

**STAGE 1: BITMAP2LCD SOFTWARE USE**

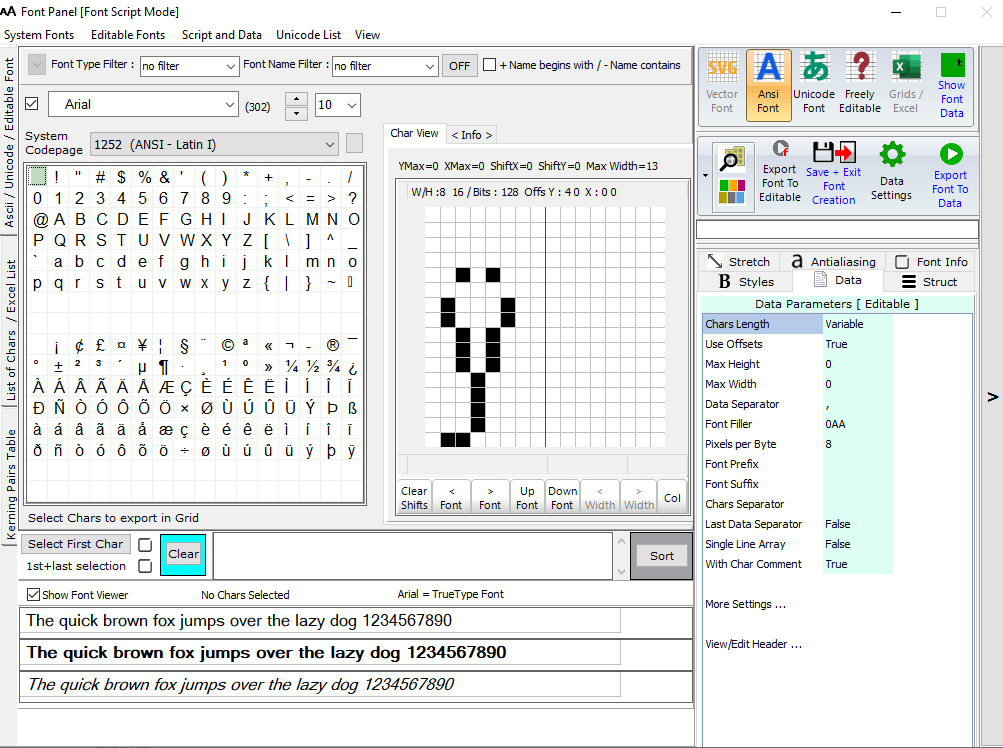
Creating the raw hex file and bin file for corresponding Font name and Font size.

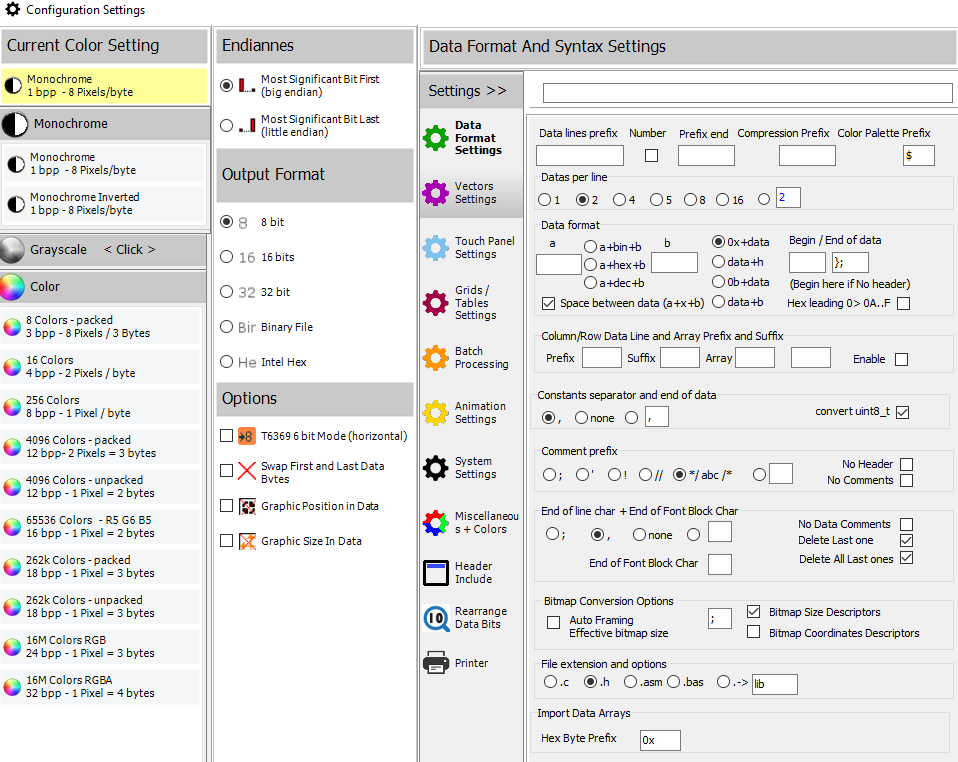
Ex: Font name : Verdana Font size: 14

1. Open the bitmap2LCD\_v4.4.exe.

2. Open Font Panel icon to select the font name and font size.

3. In font panel, click 🡪 system fonts 🡪 create /Load a GLCD fonts



4. In the configuration settings, Click 🡪 monochrome 1bpp – 8pixels/byte 🡪8-bit

5. In Font panel, Select all the 96 characters in order and click 🡪 Export font to data. It generates Verdana\_14\_1252 (ANSI - Latin I).h

6. In the configuration settings, Click 🡪 monochrome 1bpp – 8pixels/byte 🡪Binary File.

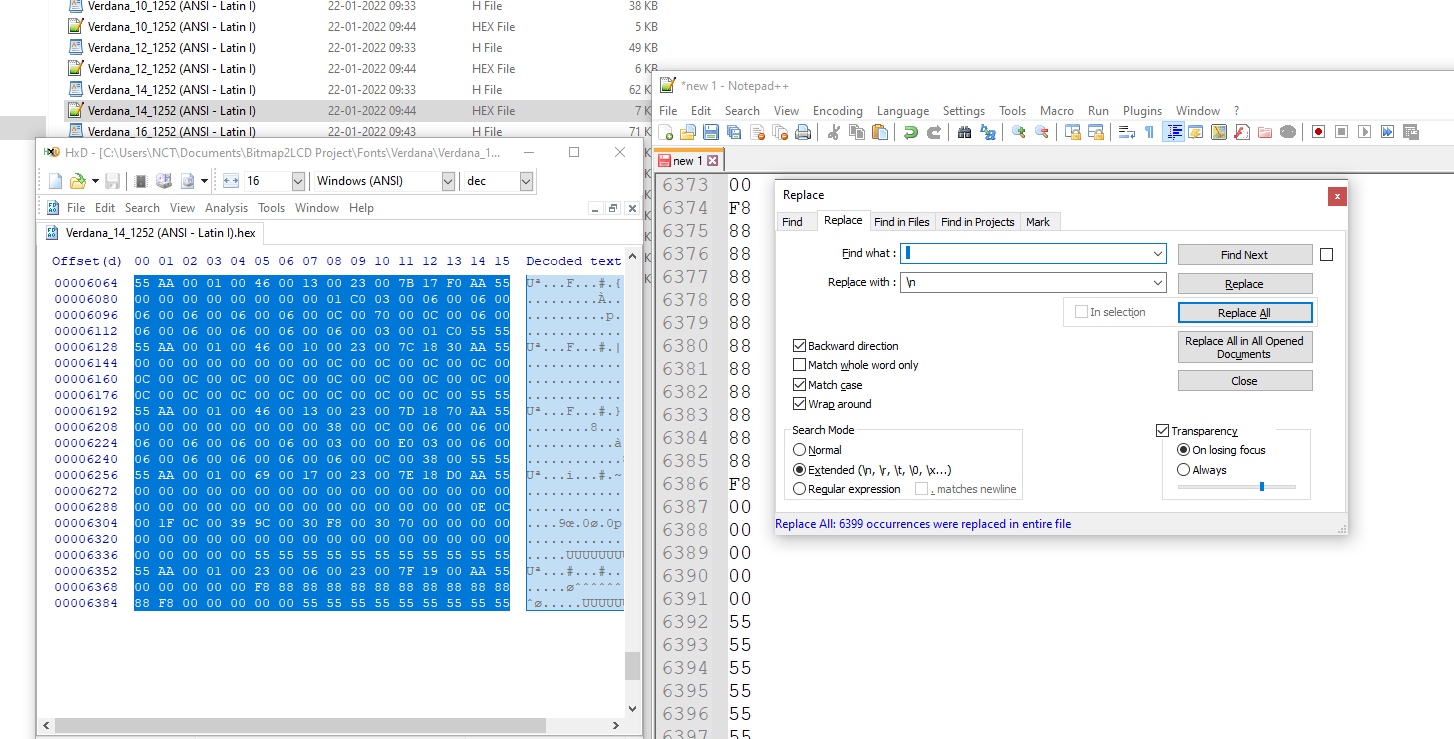
7. In Font panel, Select all the 96 characters and click 🡪 Export font to data. It generates Verdana\_14\_1252 (ANSI - Latin I).hex

8. Check the Verdana\_14\_1252 (ANSI - Latin I).h file, whether font file generates in order [ start with “ ”

To DEL character]. If doesn’t generate, again do the step 4 to 7.

10. Verdana\_14\_1252 (ANSI - Latin I).h 🡪 Open it on notepad++. This file have the character physical shape.

9. Verdana\_14\_1252 (ANSI - Latin I).hex can be open through Hex neo editor and copy all the data[ CTRL + A 🡪 CTRL + C ] and paste it on another txt/hex file in notepad++.



10. Save the filename as \*.hex or \*.bin

**STAGE 2: MODELSIM SIMULATION**

1. Use the variable\_font\_rom\_verdana folder for simulation and generating the synthesizable (decode) font\_rom file.

Ex. Here I am taking font name : Verdana font size: 14

2. In RTL folder, based on font size, different \*.v files are present. As per our example, open the font14\_verdana\_decode.v.

3. As per the logic written in the font14\_verdana\_decode.v, separate the individual character font rom data, and write it in font\_14verdana\_decode.hex file. This file has organized as below.

**CHAR LENGTH 23:**

00 23 00 06 00 23 ( # ) 00 00 00 … 00 [69 bytes]

Char Length Char Bitwidth CharAscii value Char Font data

Here “#”char length is 23 but font data length is 69. Because, to address an each char, taken the maximum length of the character in 96 chars and applied to all the character.

**Original char font rom data [Verdana\_14\_1252 (ANSI - Latin I).hex ]**

55 AA 00 01 00 23 00 06 00 23 00 27 02 20 AA 55

00 00 00 00 30 30 30 30 30 00 00 00 00 00 00 00

00 00 00 00 00 00 00 55 55 55 55 55 55 55 55 55

**Decoded char font rom data [font\_14verdana\_decode.hex]**

00 23 00 06 00 27 **00** 00 00 **00** 00 00 **00** 00 00 **00**

00 00 **30** 00 00 **30** 00 00 **30** 00 00 **30** 00 00 **30** 00

00 **00** 00 … 00 **00** 00 00 ( upto 69 datas)

**Note :** Bold datas are original datas, remaining are excess data to manage the address.

**CHAR LENGTH 46:**

00 46 00 13 00 2A ( \* ) 00 00 00 … 00 [69 bytes]

Char Length Char Bitwidth CharAscii value Char Font data

Here “\*”char length is 46 but font data length is 69. Because, to address an each char, taken the maximum length of the character in 96 chars and applied to all the character.

**Original char font rom data [Verdana\_14\_1252 (ANSI - Latin I).hex ]**

55 AA 00 01 00 46 00 13 00 23 00 2A 02 E0 AA 55

00 00 00 00 00 00 00 00 04 00 44 40 75 C0 1F 00

04 00 1F 00 75 C0 44 40 04 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 55

**Decoded char font rom data [font\_14verdana\_decode.hex]**

00 46 00 13 00 2a **00** **00** 00 **00** **00** 00 **00** **00** 00 **00**

**00** 00 **04 00** 00 **44 40** 00 **75 c0** 00 **1f 00** 00 **04 00**

…… 00 00 00 00 ( upto 69 datas)

**CHAR LENGTH 69:**

00 69 00 17 00 2B ( + ) 00 00 00 … 00 [69 bytes]

Char Length Char Bitwidth CharAscii value Char Font data

Here “+”char length is 69, so font data length is 69.

**Original char font rom data [Verdana\_14\_1252 (ANSI - Latin I).hex ]**

55 AA 00 01 00 69 00 17 00 23 00 2B 03 40 AA 55

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 01 80 00 01 80 00 01 80 00 01 80

00 01 80 00 3F FC 00 3F FC 00 01 80 00 01 80 00

01 80 00 01 80 00 01 80 00 00 00 00 00 00 00 00

00 00 00 00 00 55 55 55 55 55 55 55 55 55 55 55

**Decoded char font rom data [font\_14verdana\_decode.hex]**

00 69 00 17 00 2b 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 01 80 00 01 80 00 01

80 00 01 80 00 01 80 00 3f fc 00 3f fc 00 01 80 00 01

80 00 01 80 00 01 80 00 01 80 00 00 00 00 00 00 00

00 00 00 00 00 00

Total : 75 data’s for an individual character ( 69 font data + 6 )

Total no of data’s: 75 x 96 = 7200 data’s

Like this we have to manipulate for an individual font files.

**STAGE 3: SYNTHESIS**

**1.** Use the APPATOUCHDISPLAY\_SPI\_FONT\_verdana folder to display the char in LCD/VGA .

2. Copy the data’s in font\_14verdana\_decode.hex fileand paste it on another 14\_fontverdana.hex file manually.

3. This file located in (14\_fontverdana.hex) ,

APPATOUCHDISPLAY\_SPI\_FONT\_verdana\ input\font\_file\

In this stage, font\_14verdana\_decode.hex has 7200 data’s and 14\_fontverdana.hex has 8192 data’s. Now append the extra zero’s above 7200.

In 14\_fontverdana.hex, First 6 data’s manually we have to change the hexadecimal values.

00 00

23 17h ( rowcnt )

00 00

08 08 ( don’t change the hex value)

00 00

75 4Bh ( Max\_char\_length)

These values are represent the rowcnt and max\_char\_length of the font file.These values used to calculate the address of the incoming char in ..\APPATOUCHDISPLAY\_SPI\_FONT\_verdana\rtl\image\_config\font\txt\_overlay\_font.v

14\_fontverdana.hex data’s must be in the range of multiple of 2048( Flash page size). That’s why extra zero’s appended.

4. Change the font file name in char\_config\_rom.v and do synthesis will get the font in display…….