



Stellaris Graphic lib and Ethernet application hands on

StellarisWare – Graphics Library

- Written entirely in C, easy-to-use, efficient.
- Three layers of functionality, each directly callable:

- ✓ Display Driver Layer (Lowest Level)
- ✓ Graphics Primitives Layer ...
- ✓ Widget Layer (Highest Level)

- **Graphics Primitives:**

- ✓ Point, Line, Rectangle, Circle, Font, Image, Context, Buffer
- ✓ 134 Computer Modern predefined fonts available
- ✓ Support for 24-bit color

- **Widgets:**

- ✓ Canvas, Checkbox, Container, Push Button, Radio Button, Slider, ListBox

- **Special Utilities**

- ✓ *frasterize*: render your own font
- ✓ *Imi-button*: predefined button with shadow and 3-D
- ✓ *pnmtoc*: Convert image file to GraphicsLib format



Primitives



Radio Buttons



Checkbox



Container



Canvas



Push Buttons

Graphic Hands On Prepare

- **This example is to put photo on the LCD**
- **Find a JPG file and stretch it to 65x55**
- **Put JPG file to the folder \StellarisWare\tools\bin**
- **Install GIMP and then open GIMP**
- Load the file (File->Open).
- Convert the image to indexed mode (Image->Mode->Indexed).
Select “Generate optimum palette” and select either 2, 16, or 256 as the maximum number of colors (for a 1 BPP, 4 BPP, or 8 BPP image respectively). If the image is already in indexed mode, it can be converted to
RGB mode (Image->Mode->RGB) and then back to indexed mode.
- Save the file as a PNM image (File->Save As). Select raw format when prompted.
- Use pnmtoc to convert the PNM image into a C array.

PNMTOC

- Pre-compiled executable file locate in \StellarisWare\tools\bin\
- Put your pnm file to this directory too
- Command DOS runs to this directory
- Type the command :
pnmtoc -c file<your file name>.pnm >output.c
- You will get the table like this :

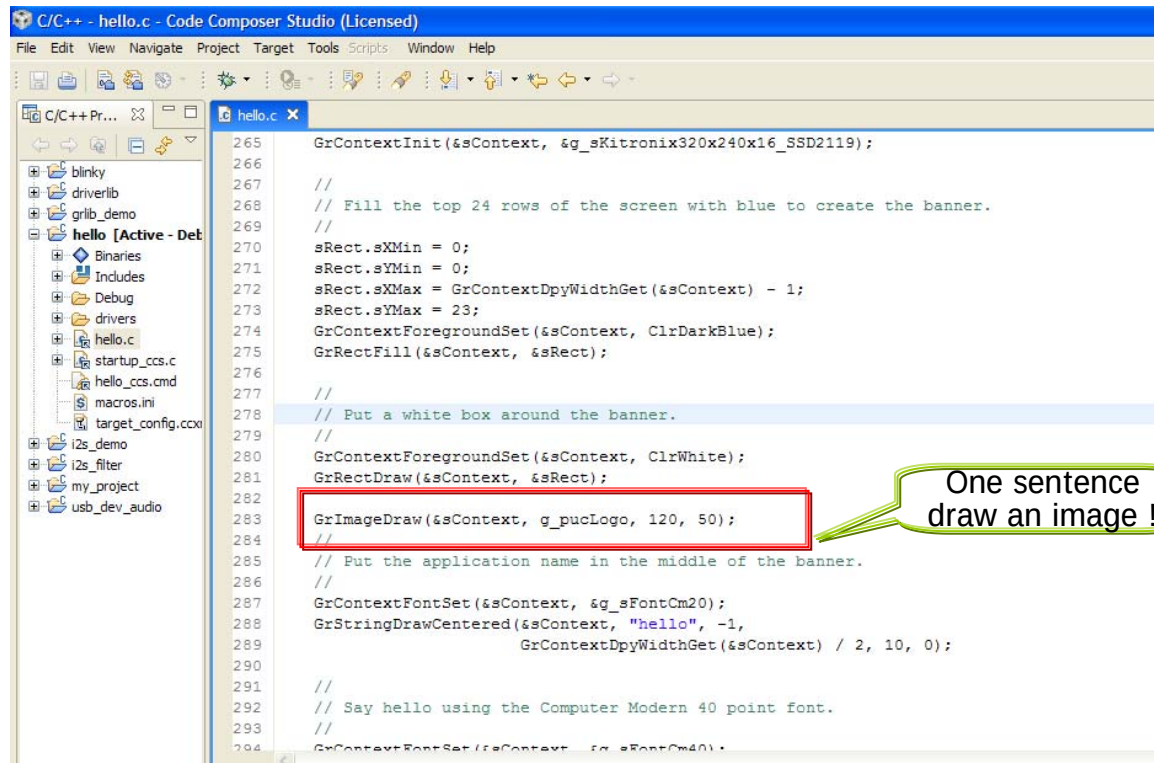
```
const unsigned char g_pucImage[] =  
{  
    IMAGE_FMT_4BPP_COMP,  
    65, 0, Image size  
    55, 0,  
    Color Depth 15,  
    0x16, 0x0f, 0x0b,  
    0x09, 0x14, 0x13,  
    0x1a, 0x2f, 0x2c,  
    0x2b, 0x2e, 0x29,  
    0x29, 0x54, 0x57,  
    0x47, 0x56, 0x56,  
    0x7e, 0x62, 0x51,  
    0xd5, 0x9c, 0x6b,  
    0x69, 0x95, 0xa1,  
    0x90, 0x9a, 0xa2,  
    0xb7, 0xa2, 0x92,  
    0xec, 0xca, 0xae,  
    0xb9, 0xcb, 0xd6,  
    0xd6, 0xd0, 0xcb,  
    0xe0, 0xee, 0xf4,  
    0xf1, 0xef, 0xec,  
  
    0x00, 0xdf, 0xa6, 0x66, 0x67, 0x67, 0x66, 0xff, 0xbf, 0x00, 0xff, 0xff
```



Copy This table to the Image.c and recompile the project

Graphic Demo

- Import project to your workspace
\\StellarisWare\boards\dk-lm3s9b96\hello\
- Place the content of table g_pucLogo[] with what you get by PNMTOC
- Draw the image on the screen



```
265 GrContextInit(&sContext, &g_sKitronix320x240x16_SSD2119);
266 //
267 // Fill the top 24 rows of the screen with blue to create the banner.
268 //
269 sRect.sXMin = 0;
270 sRect.sYMin = 0;
271 sRect.sXMax = GrContextDpyWidthGet(&sContext) - 1;
272 sRect.sYMax = 23;
273 GrContextForegroundSet(&sContext, CClrDarkBlue);
274 GrRectFill(&sContext, &sRect);
275 //
276 // Put a white box around the banner.
277 //
278 GrContextForegroundSet(&sContext, CClrWhite);
279 GrRectDraw(&sContext, &sRect);
280 //
281 GrImageDraw(&sContext, g_pucLogo, 120, 50);
282 //
283 // Put the application name in the middle of the banner.
284 //
285 GrContextFontSet(&sContext, &g_sFontCm20);
286 GrStringDrawCentered(&sContext, "hello", -1,
287                     GrContextDpyWidthGet(&sContext) / 2, 10, 0);
288 //
289 // Say hello using the Computer Modern 40 point font.
290 //
291 GrContextFontSet(&sContext, &g_sFontCm40);
```

G_pucLogo contains what you get from PNMTOC

