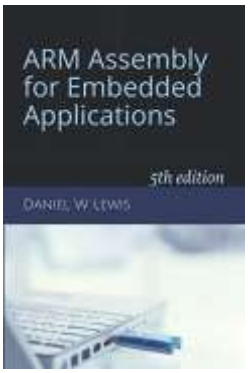

Programming Lab 9E

Image Morphing

Topics: Floating-point arithmetic and conversions, bit manipulation, loops

Click to download Lab9E-Main.c

Prerequisite Reading: Chapters 1-9
Revised: June 22, 2021

Background*[1]*: Morphing is a special effect used in motion pictures and animations that changes (or morphs) one image into another through a seamless transition. Each image is a two-dimensional array of pixels, with each pixel defined as the combination of three unsigned integers, one each to represent the intensity of the pixel's red, blue and green color components. During morphing, a blend of two images is displayed at each stage of the transition, accomplished by combining corresponding pixels from the two images using a percentage of the component values from each pixel in the first image with a percentage of the component values from the corresponding pixels in the second image such that the percentages total 100%. For example, to blend 25% of the color of one pixel (`pxl1`) with 75% of the color of another (`pxl2`), the color components of the displayed pixel would be computed as:


```
displayed.red = 0.25 * pxl1.red + 0.75 * pxl2.red ;
displayed.blu = 0.25 * pxl1.blu + 0.75 * pxl2.blu ;
displayed.grn = 0.25 * pxl1.grn + 0.75 * pxl2.grn ;
```

Assignment: The main program will compile and run without writing any assembly. However, your task is to create an equivalent replacement in assembly language for function `BlendPixels` found in the C main program and whose interface is defined below. The original C version of the function has been defined as "weak" so that the linker will automatically replace it in the executable image by the one you create in assembly; you do not need to remove the C version.

```
uint32_t BlendPixels(float percent, uint32_t pxl1rgb, uint32_t pxl2rgb) ;
```

`percent`	A number between 0.0 and 1.0
`pxl1rgb`	A word containing the red, blue and green components of a pixel from the first image.
`pxl2rgb`	A word containing the red, blue and green components of a pixel from the second image.

The red, green and blue components of each pixel are stored in bits 23-16, 15-8, and 7-0 respectively.

Test your implementation of the `BlendPixels` function using the C main program. If your code is correct, the display should display a blend of two images as shown above. Use the slider to change the blend from the first image to the second. The two original images are viewable at either end of the slider.

[1] https://en.wikipedia.org/wiki/Morphing