USING STEGANOGRAPHY TO APPLY BITWISE OPERATORS AND FILE PROCESSING IN C *

NIFTY ASSIGNMENT

Jim Kirk
Union University, Jackson, TN
(731) 661-5274
jkirk@uu.edu

ASSIGNMENT OBJECTIVES

The primary objective is to illustrate and practice the use of bitwise operators and binary file processing. A secondary objective is to introduce the idea of file structures and examine the structure of one version of the device-independent bitmap (DIB). The assignment is used in a 2nd programming course for CS majors and Engineering majors.

ASSIGNMENT PREPARATION

Students are introduced to the syntax and semantics of the bitwise operators with several simple examples. The difference between with bitwise operators and logical operators is discussed. The applications of bit masking, querying the status of a bit, and toggling bits are introduced briefly. This assignment also serves to review and reinforce students' understanding of file input/output.

IN-CLASS ASSIGNMENT

Students are given a 24-bit uncompressed DIB file in which the color data has been modified by overwriting the low nybble of each byte with the high nybble of the corresponding pixel color from another image file of the same size. In effect, the modified DIB contains the most significant color information from 2 images. Because only the 4 least-significant bits of each byte have been overwritten, there is a maximum loss of color information of $15/255 < 5.9\%$. Students open the image with any image viewer and examine it. They are not told, and they cannot detect that the DIB has been modified.

Students are shown a diagram of the structure of a 24-bit uncompressed DIB file. They are told that we are going to test some bitwise operators together, and that the

---

instructor will lead the class in writing a program that will flip the nybbles of each byte in the DIB image color data. The steps in this process may be found below, with sample code at http://www.ccsc-ms.org/nifty/14/kir/.

1. Create a new binary file for writing and open the modified DIB for binary reading.
2. Seek to the 10th byte in the DIB (the end of the bitmap file header) and read an integer n. This integer will be the offset in bytes from the beginning of the file to the image color data.
3. Copy the file headers (the first n bytes) from from the input to the output file.
4. From the start of the color data at byte n to the end of the DIB, read each byte, reverse the nybbles of each, and write the reversed byte to the output.

Students individually finish and execute the program. Upon viewing the new output DIB, it appears that reversing the nybbles has produced an entirely new image, with no evidence of the original image. Students' outbursts around the lab can be entertaining, my favorite being: "What magic is this?!"

HOMEWORK

Motivated students are usually interested in the program that produced the dual-image DIB. Writing this program can be a good short homework project without much additional preparation. With a little more preparation on bitwise operators, students can write their own programs that replace low order bits from each byte with text data, allowing them to encode messages within images. Our most sophisticated attempt has been a program that, given the text message and the image, first determines the smallest number of bits per pixel required to encode the message and then encodes it in the image.