**CS 3308 Note 3-2**

**Packing And UnPacking Red Green Blue**

**Using Shifts and Masks**

The following code uses a struct and two methods to pack and unpack 8 bit red green, and blue values into a single unsigned integer

Question: why is the static cast needed below?

#include <iostream>

using namespace std;

typedef struct RGB {

unsigned int s, b, g, r;

}RGB;

unsigned int packRGB(const RGB \*rgb) {

return rgb->s << 24 | rgb->b << 16 | rgb->g << 8 | rgb->r;

}

RGB unpackRGB(unsigned int rgb) {

RGB unpacked = {rgb >> 24 & 0XFF, rgb >> 16 & 0XFF, rgb >> 8 & 0XFF, rgb & 0XFF};

return unpacked;

}

int main() {

RGB rgb = {'a', 'b','c', 'd'};

// rgb->s = 97, rgb->b = 98, rgb->g = 99, rbg->r = 100;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Byte3 | Byte2 | Byte1 | Byte0 |
| s | 0 | 0 | 0 | 97 |
| s << 24 | b << 16 | 97  01100001 | 0  00000000 | 0  00000000 | 0  00000000 |
| b | 0  00000000 | 0  00000000 | 0  00000000 | 98  01100010 |
| b << 16 | 0  00000000 | 98 01100010 | 0  00000000 | 0  00000000 |
| g | 0  00000000 | 0  00000000 | 0  00000000 | 99  01100011 |
| g << 8 | 0  00000000 | 0  00000000 | 99  01100011 | 0  00000000 |
| r | 0  00000000 | 0  00000000 | 0  00000000 | 100  01100100 |
| Unpacked | 97  01100001  00000000 | 98 01100010  00000000 | 99  01100011  00000000 | 100  01100100  11111111 |

unsigned int packed\_rgb = packRGB(&rgb);

void \*pVoid = &packed\_rgb;

char \*a = static\_cast<char\*> (pVoid);

cout << "---" << a[0] << "---" << endl; // can access \*a as an array

cout << "---" << a[1] << "---" << endl;

cout << "---" << a[2] << "---" << endl;

cout << "---" << a[3] << "---" << endl;

cout << endl;

for(int i=0;i<4;i++) // can access \*a with pointer arithmetic

{

cout << \*a << " ";

a++;

}

cout << endl << endl;

RGB unpacked\_rgb = unpackRGB(packed\_rgb);

printf("Packed RGB = %u\n", packed\_rgb);

printf("Unpacked RGB = (%u, %u, %u, %u)", unpacked\_rgb.r, unpacked\_rgb.g, unpacked\_rgb.b, unpacked\_rgb.s);

return 0;

}