LE/EECS 2021 4.00   Computer Organization

LAB A REPORT

Michael Williams 211087798

Section E

September 22 2014

The work in this report is my own. I have read and understood York University academic dishonesty policy and I did not violate the senate dishonesty policy in writing this report.

ABSTRACT

In this lab, I was introduced to binary code and some of the many methods available to manipulate it on a bit level. The purpose was to familiarize us with some of the bit manipulation that we will be undergoing in future labs. We were taught to convert values into binary and hexadecimal, compare values and produce a comparative result, directly manipulate individual bits of our choice, and to use bit manipulation to do arithmetic. In conclusion, these programs provided methods to manipulate even the bits themselves of a given input, and change numbers entirely, giving us knowledge for our future labs a little more insight into how bits function.

EQUIPMENT

-laptop

-java eclipse

-putty

-xming

METHODS/PROCEDURES

For the most part, the lab was a simple. For programs 1-5, I simply followed the given instructions, and made little alterations to what was provided. For programs 6-8, I did a little more work. For the most part, this meant creating masks for the specific bit manipulation needed for the program, as well as coding that would provide me the ability to change certain bits in given values. One noteworthy implementation was in program 7, shown below. In order to decide what type of bit manipulation was to be used, it was necessary to create an if statement to determine this, after isolating the bit to be changed. This was probably the only trouble I ran into throughout the lab. The following describes isolating the needed bit, and then creating appropriate masks and processes to replace a given bit with the opposite value, at bits 20 and 10 respectively, within program 7.

**if**(y == 0){

mask = Integer.*parseInt*("1111111111101111111111111111111",2); //all but bit #20 = 1

x = x & mask; // forcing 0 into bit#20

}//if y ==0

**else**{

mask = Integer.*parseInt*("00000000000100000000000000000000",2); //all but bit #20 = 0

x = x | mask; // forcing 1 into bit#20

}//if y ==

**if**(z == 0){

mask = 1024;

mask = ~mask; //all but bit #10 = 1

x = x & mask; // forcing 0 into bit#10

}//if x ==0

**else**{

mask = 1024; //all but bit #10 = 0

x = x | mask; // forcing 1 into bit#20

}//if x == 1

RESULTS

The results were as expected throughout all the labs. For labs 1-5, and 8 this was simply the results described within the program instructions. For labs 6-7, the results were variable, as they were dependent on the output. However, the output did successfully result in accurate responses. For example, in program 6, when using the value 7841 as input, the result was the binary sequence, such that bit 10 was given the value 1, and bit 11 was given the value 0, in this case being 1011010100001. Another example is in program 7, where the given task was to switch bits 10 and 20 with one another. When using the input 9431, the expected results were 100000010000011010111.

DISCUSSION

There wasn’t much to discuss in terms of this lab, considering this was the introductory lab. For the most part, everything was straight forward. The only part I had little trouble with was isolating methods to change bit values, but once I figured out how to implement masks correctly, the issue became easily correctable.

CONCLUSION

In conclusion, the lab was very straight forward, and did not provide much challenge in terms of understanding, or implementation. I achieved everything I sought to learn with this lab, and easily overcame any issues that arised while making the programs, while learning about bit manipulation, the many processes it involves, and the drastic changes simply changing a single bit can have to given values.