NEW YORK CITY COLLEGE OF TECHNOLOGY

THE CITY UNIVERSITY OF NEW YORK

300 JAY STREET, BROOKLYN, NY 11201-1909

Department of Computer Engineering Technology

CET 3510 - OL25 - Microcomputer Systems Technology lab

LABORATORY REPORT No_08

Lab_No_08

Professor: Dr. Rubén Velásquez, Ph.D.

Spring 2021

Student's Name: Ashahi Shafin

Student's ID: 23607352

Preparation date: *April 14, 2021*

Due date: *April 21, 2021*

1. Table of content

2. OBJECTIVE

Laboratory No 8. Write a C/C++ program to exam logic instructions for the processor, such as AND(&), OR(|), XOR(^) and NOT(~). The AND, OR, XOR, and NOT instructions perform logic functions on a byte, word or doubleword stored in a register or memory location.

3. LABORATORY TOOLS TO PERFORM THE TASK

Computer NZXT

Microsoft visual studio 2015 Software
Microsoft Word 2019 Software
Microsoft Notepad Software

4. SOURCE CODE

Laboratory No_08 Example 8.1

Source Code

```
#include <iostream>
#include <bitset>
#include "windows.h"
using namespace std;

void mainMenu();
void subMenu1();
void AND_operation(unsigned short r1, unsigned short r2);
void OR_operation(unsigned short r1, unsigned short r2);
void XOR_operation(unsigned short r1, unsigned short r2);
void NOT_operation(unsigned short r1);


int main()
{
	printf(" Lab_No_08_Logic_Operations\n");
	printf(" Module: C++ progrmming \n");
	printf(" Ashahi Shafin, ID#23607352\n");
	printf(" CET3510-OL25\n");
	printf(" Presentation date: April 14, 2021\n");
	printf(" Due date: April 21, 2021\n");
	printf(" Example 8.1 logicOperations.cpp\n");
	printf(" file name: 8.1 logicOperations.cpp\n");
	printf("-------------------------------------------------------------------------------
\n");

	//Declare variables here
```

```
	char ch, ch1, ch2, ch3;
	unsigned short r1, r2;			// A 16-bit word declaration

	cout << "Start the logic operation Y/N, enter Y(y) or N(n): ";
	cin >> ch;
	ch1 = ch;

	mainMenu();
	cout << "Menu Options: ";
	cin >> ch;
	ch2 = ch;
	while (ch1 == 'Y' || ch1 == 'y')
	{
		if (ch2 == '1')
		{
		Submenu:
			subMenu1;
			cout << "Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): ";
			std::cin >> ch;
			ch3 = ch;
			switch (ch3)
			{
			case 'a':
			{
				cout << "Enter the first number in hexadecimal format (for example,
12ae): ";
				cin >> hex >> r1;
				cout << "Enter the second number in hexadecimal format (for example,
ff00): ";
				cin >> hex >> r2;
				AND_operation(r1, r2);
				break;
			}
			case 'b':
			{
				cout << "Enter the first number in hexadecimal format (for example,
12ae): ";
				cin >> hex >> r1;
				cout << "Enter the second number in hexadecimal format (for example,
ff00): ";
				cin >> hex >> r2;
				OR_operation(r1, r2);
				break;
			}
			case 'c':
			{
				cout << "Enter the first number in hexadecimal format (for example,
12ae): ";
				cin >> hex >> r1;
				cout << "Enter the second number in hexadecimal format (for example,
ff00): ";
				cin >> hex >> r2;
				XOR_operation(r1, r2);
				break;
			}
			case 'd':
			{
				cout << "Enter the first number in hexadecimal format (for example,
12ae): ";
```

```
                        cin >> hex >> r1;
                        NOT_operation(r1);
                        break;
                }

                default: goto Submenu;

                }
        }
        else
        {
                goto EndLable;
        }

        cout << "Do you like to continue the logic operations (Y/N)? Enter Y(y) or N(n): "
<< endl;
        cin >> ch;
        ch1 = ch;
    }

EndLable:
    cout << "Exit program" << endl;

    system("pause");
    exit(0);
    return 0;
}
void mainMenu()
{
    cout << "Menu:" << endl;
    cout << "1, Perform logic operation with 16-bit operand(s)" << endl;
    cout << "2, Exit" << endl;
}

void subMenu1()
{
    cout << "a. Input two 16-bit unsigned number operands and perform an AND (&)
operation.\n"
        << "   Display the operands and the result in binary format. " << endl;
    cout << "b. Input two 16-bit unsigned number operands and perform an OR (|) operation.\n"
        << "   Display the operands and the result in binary format. " << endl;
    cout << "c. Input two 16-bit unsigned number operands and perform an XOR (^)
operation.\n"
        << "   Display the operands and the result in binary format. " << endl;
    cout << "d. Input two 16-bit unsigned number operands and perform an NOT (~)
operation.\n"
        << "   Display the operands and the result in binary format. " << endl;
}

void AND_operation(unsigned short r1, unsigned short r2)
{
    unsigned short r;

    // AND operation
    _asm
    {
    mov AX, r1;
    mov CX, r2;
    and AX, CX;
```

```
        mov r, AX;
        }


        bitset<16> operant1_Bits(r1);
        bitset<16> operant2_Bits(r2);
        bitset<16> result_Bits(r);

        cout << "Perform an AND operation:" << endl;
        cout << "\t\t" << operant1_Bits << endl;
        cout << "\tAND" << "\t" << operant2_Bits << endl;
        cout << "-----------------------------------\n";
        cout << "\t\t" << result_Bits << endl;
        cout << "============================================================\n";
}

void OR_operation(unsigned short r1, unsigned short r2)
{
        unsigned short r;

        // OR operation
        _asm
        {
                mov AX, r1;
                mov CX, r2;
                or AX, CX;
                mov r, AX;
        }

        bitset<16> operant1_Bits(r1);
        bitset<16> operant2_Bits(r2);
        bitset<16> result_Bits(r);

        cout << "Perform an OR operation:" << endl;
        cout << "\t\t" << operant1_Bits << endl;
        cout << "\tOR" << "\t" << operant2_Bits << endl;
        cout << "-----------------------------------\n";
        cout << "\t\t" << result_Bits << endl;
        cout << "============================================================\n";
}

void XOR_operation(unsigned short r1, unsigned short r2)
{
        unsigned short r;

        // XOR operation
        _asm
        {
                mov AX, r1;
                mov CX, r2;
                xor AX, CX;
                mov r, AX;
        }

        bitset<16> operant1_Bits(r1);
        bitset<16> operant2_Bits(r2);
        bitset<16> result_Bits(r);

        cout << "Perform an XOR operation:" << endl;
```

```
        cout << "\t\t" << operant1_Bits << endl;
        cout << "\tXOR" << "\t" << operant2_Bits << endl;
        cout << "-----------------------------------\n";
        cout << "\t\t" << result_Bits << endl;
        cout << "============================================================\n";
}

void NOT_operation(unsigned short r1)
{
        unsigned short r;

        // NOT operation
        _asm
        {
                mov AX, r1;
                not AX;
                mov r, AX;
        }

        bitset<16> operant1_Bits(r1);
        bitset<16> result_Bits(r);

        cout << "Perform an XOR operation:" << endl;
        cout << "\tNOT" << "\t" << operant1_Bits << endl;
        cout << "-----------------------------------\n";
        cout << "\t\t" << result_Bits << endl;
        cout << "============================================================\n";
}
```

5. *SOURCE CODE LINE DESCRIPTION*

Laboratory No_08 Example 8.1

Line	***Source Code Description***
01	*#include "stdafx.h"*
Precompiled Header **stdafx**. **h** is basically used in Microsoft Visual Studio to let the compiler know the files that are once compiled and no need to compile it from scratch. ... **h**" before this **includes** then the compiler will find the compiled header files from **stdafx**. **h** and does not compiled it from scratch.	
02	*#include "stdio.h"*
The stdio.h (standard library header) is a file with ".h" extension that contains the prototypes of standard input-output functions used in c.	
03	#include<iostream>
h, **iostream** provides basic input and output services for C++ programs. **iostream uses** the objects cin ,	

cout , cerr , and clog for sending data to and from the standard streams input, output, error (unbuffered), and log (buffered) respectively.

04

```
int main(void)
{
MAIN PROGRAM
}
```

In C and C++ **int main**(**void**) **means** that the function takes NO arguments. ... **Int main**(**void**) is used in C to restrict the function to take any arguments, if you do not put **void** in those brackets, the function will take ANY number of arguments you supply at call.

05

```
char ch, ch1, ch2, ch3;
unsigned short r1, r2;
```

A **variable** is a name given to a memory location. It is the basic unit of storage in a program. The value stored in a **variable** can be changed during program execution.

06

```
cout << "Start the logic operation Y/N, enter Y(y) or N(n): ";
cout << "Menu Options: ";
```

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

07

```
while (ch1 == 'Y' || ch1 == 'y')
```

A **while** loop statement repeatedly executes a target statement as long as a given condition is true.

08

```
if (ch2 == '1')
```

The **if** statement allows you to control **if** a program enters a section of code or not based on whether a given condition is true or false. One of the important functions of the **if** statement is that it allows the program to select an action based upon the user's input.

09

```
cout << "Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): ";
```

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

10

```
switch (ch3)
```

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

11

```
case 'a':
```

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

12

```
cout << "Enter the first number in hexadecimal format (for example, 12ae): ";
cin >> hex >> r1;
cout << "Enter the second number in hexadecimal format (for example, ff00): ";
cin >> hex >> r2;
```

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

13	`case 'b':`

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

14	`cout << "Enter the first number in hexadecimal format (for example, 12ae): ";`
	`cin >> hex >> r1;`
	`cout << "Enter the second number in hexadecimal format (for example, ff00): ";`
	`cin >> hex >> r2;`

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

15	`case 'c':`

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

16	`cout << "Enter the first number in hexadecimal format (for example, 12ae): ";`
	`cin >> hex >> r1;`
	`cout << "Enter the second number in hexadecimal format (for example, ff00): ";`
	`cin >> hex >> r2;`

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

17	`case 'd':`

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

18	`cout << "Enter the first number in hexadecimal format (for example, 12ae): ";` `cin >> hex >> r1;`

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

19	`default: goto Submenu;`

A **default** argument is a value provided in a function declaration that is automatically assigned by the compiler if the caller of the function doesn't provide a value for the argument with a **default** value.

20	`else`

Use **else** to specify a block of code to be executed, if the same condition is false. Use **else** if to specify a new condition to test, if the first condition is false.

21	`cout << "Exit program" << endl;`

The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

22	`system("pause");`

The **system**() function is a part of the C/**C++** standard library. It is used to pass the commands that can be executed in the command processor or the terminal of the operating **system**, and finally returns the command after it has been completed.

23	`exit(0);`
The **exit** function, declared in <stdlib. h>, terminates a **C++** program.	
24	`return 0;`
Terminates the execution of a function and **returns** control to the calling function (or to the operating system if you transfer control from the main function). Execution resumes in the calling function at the point immediately following the call.	
25	`void mainMenu()`
When used as a function return type, the **void** keyword specifies that the function does not return a value. When used for a function's parameter list, **void** specifies that the function takes no parameters. When used in the declaration of a pointer, **void** specifies that the pointer is "universal."	

27	`cout << "Menu:" << endl;`
	`cout << "1, Perform logic operation with 16-bit operand(s)" << endl;`
	`cout << "2, Exit" << endl;`

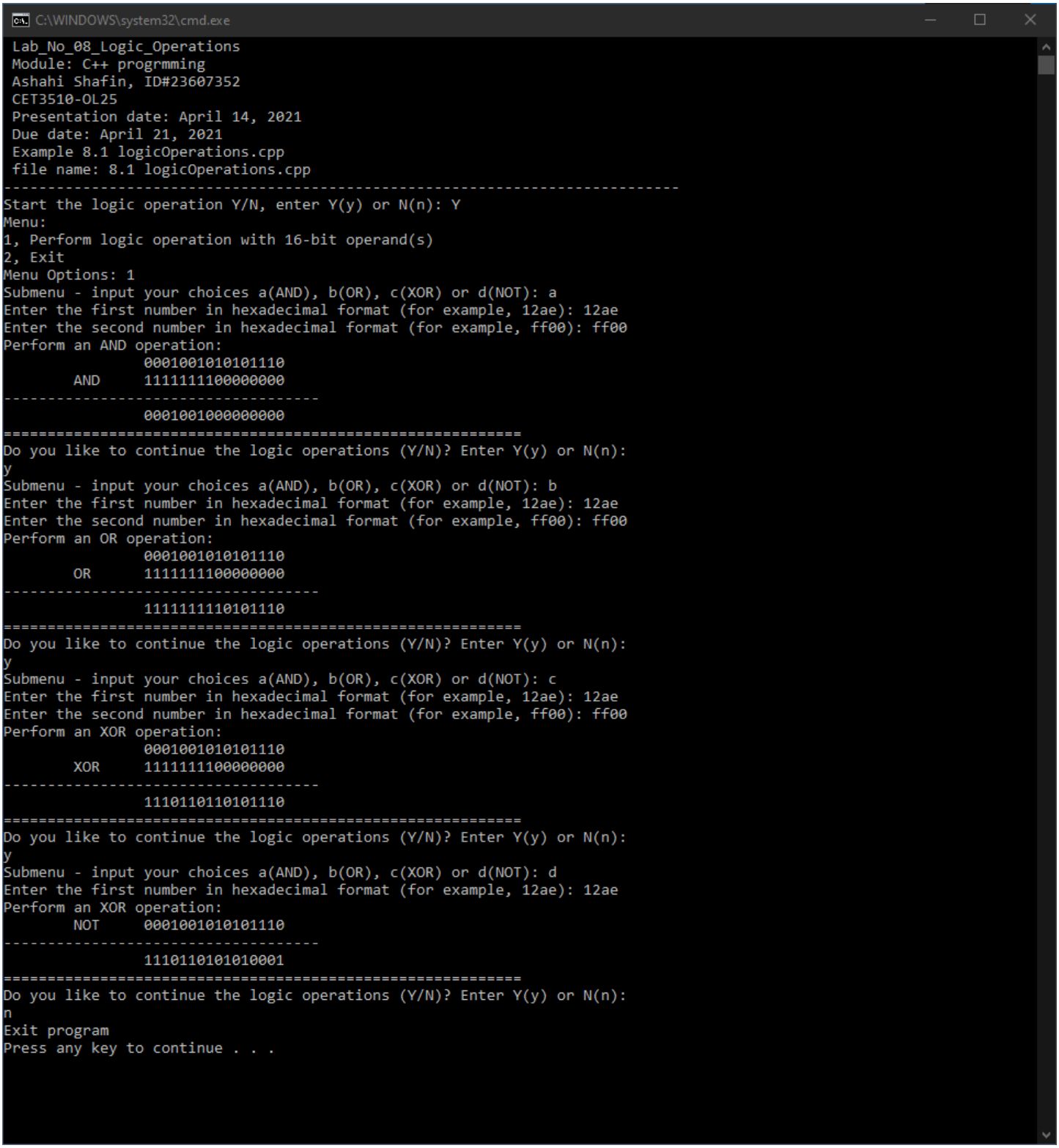
The **cout** object in **C++** is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.	
28	`void subMenu1()`
When used as a function return type, the **void** keyword specifies that the function does not return a value. When used for a function's parameter list, **void** specifies that the function takes no parameters. When used in the declaration of a pointer, **void** specifies that the pointer is "universal."	

29	`cout << "a. Input two 16-bit unsigned number operands and perform an AND (&) operation.\n"`
	`<< "   Display the operands and the result in binary format. " << endl;`
	`cout << "b. Input two 16-bit unsigned number operands and perform an OR (|) operation.\n"`
	`<< "   Display the operands and the result in binary format. " << endl;`
	`cout << "c. Input two 16-bit unsigned number operands and perform an XOR (^) operation.\n"`
	`<< "   Display the operands and the result in binary format. " << endl;`
	`cout << "d. Input two 16-bit unsigned number operands and perform an NOT (~) operation.\n"`
	`<< "   Display the operands and the result in binary format. " << endl;`

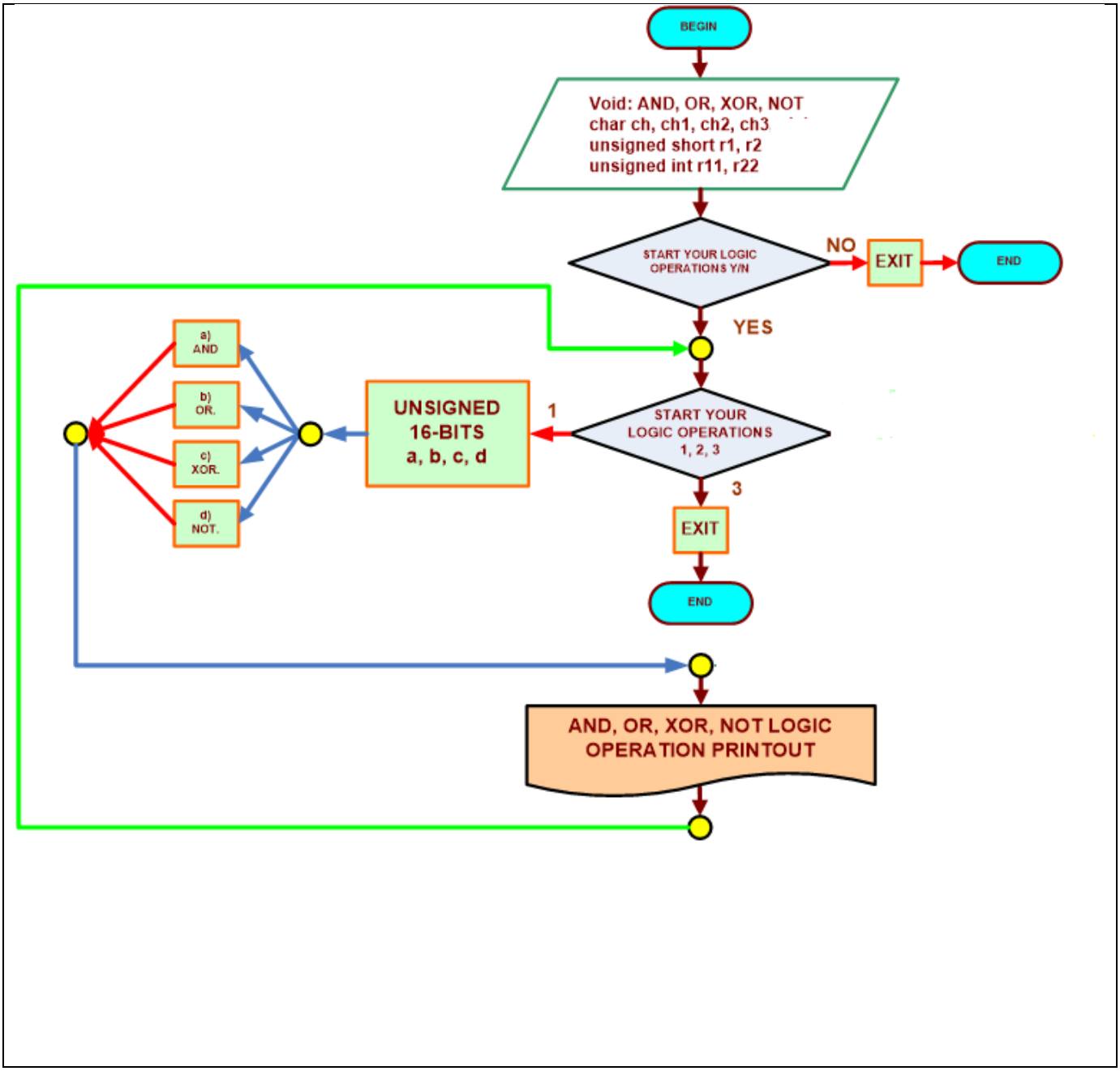
6. EXPLANING OUTPUT RESULTS

Laboratory No_08 Example 8.1

i. This is the output console of example 8.1 for laboratory 8. It shows AND OR NOR NOT outputs with the 12ae and ff00 in the hexadecimal value when they are moved in the registers.


C:\WINDOWS\system32\cmd.exe
Lab_No_08_Logic_Operations
Module: C++ progrmming
Ashahi Shafin, ID#23607352
CET3510-OL25
Presentation date: April 14, 2021
Due date: April 21, 2021
Example 8.1 logicOperations.cpp
file name: 8.1 logicOperations.cpp
Start the logic operation Y/N, enter Y(y) or N(n): Y
Menu:
1, Perform logic operation with 16-bit operand(s)
2, Exit
Menu Options: 1
Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): a
Enter the first number in hexadecimal format (for example, 12ae): 12ae
Enter the second number in hexadecimal format (for example, ff00): ff00
Perform an AND operation:
0001001010101110
AND 1111111100000000
0001001000000000
Do you like to continue the logic operations (Y/N)? Enter Y(y) or N(n):
y
Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): b
Enter the first number in hexadecimal format (for example, 12ae): 12ae
Enter the second number in hexadecimal format (for example, ff00): ff00
Perform an OR operation:
0001001010101110
OR 1111111100000000
1111111110101110
Do you like to continue the logic operations (Y/N)? Enter Y(y) or N(n):
y
Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): c
Enter the first number in hexadecimal format (for example, 12ae): 12ae
Enter the second number in hexadecimal format (for example, ff00): ff00
Perform an XOR operation:
0001001010101110
XOR 1111111100000000
1110110110101110
Do you like to continue the logic operations (Y/N)? Enter Y(y) or N(n):
y
Submenu - input your choices a(AND), b(OR), c(XOR) or d(NOT): d
Enter the first number in hexadecimal format (for example, 12ae): 12ae
Perform an XOR operation:
NOT 0001001010101110
1110110101010001
Do you like to continue the logic operations (Y/N)? Enter Y(y) or N(n):
n
Exit program
Press any key to continue . . .

7. PROGRAM FLOW CHART:



8. COMMENTS

Comments on Outputs and Tables is that code shows how it want sub menus and how they work and they also use 16 bits to find the hex value. It also shows how the MOV can used to find the hex value

9. CONCLUSION

In this experiment, the module shows when we use variable to ADD OR NOR NOT it can them into hex values this will helps when you are trying to fine the new value in that form. The par of this lab was making the code because you had to make spread code for ADD OR NOR NOT so you can get values for them and another thing was hard was make the sub menus because they were big

A real-life application for laboratory No 7 would be in a computer because computer they are always moving files and sharing them and when this is happen they have transfer 16 at the same time and this very useful with servers because they need to transfer files and connect with each other's.