|  |  |
| --- | --- |
| The UWindsor Logo | University of Windsor  Faculty of Science  School of Computer Science | COMP-2650  Computer Architecture I: Digital Design  Winter 2022 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Lab# | Date | Title | | Due Date | Grade Release Date |
| Lab 11 | Week 11 | **Gray Code** | April 06, 2022, Wednesday 4 AM EDT | | April. 11, 2022 |

This lab's objectives will be to master the topics in logic circuit design by implementing the algorithms with a programming language, herein, C/C++.

**Step 1. Environment Setup**

Our programming environment is the same as the first lab (Lab 01). In this lab, we want to implement the Gray code, named after Frank Gray[[1]](#footnote-1). This code is an ordering of the binary numeral system such that two successive values differ in only one bit. For example, the Gray codes for the decimal numbers 4 and 5 are 0110 and 0111, where the only change is the first bit. To implement Gray code, we must first convert a given decimal number to a binary number. Then, we follow the below steps:

Moving from the highest significant bit to the lowest significant bit (last bit to the first bit)

1. The last bit of the Gray code is the same as the last bit of the binary number
2. The i-th bit of the Gray code is the XOR of the i-th and (i+1)-th bits of the binary number

For instance, the Gray code for 20 is:

Chart, bar chart

Description automatically generated

**Lab Assignment**

You should implement the above algorithm output a menu of commands as follows:

Enter the encoding command number:

1. Exit
2. Gray code

If a user selects (1), the program asks for a decimal number:

Enter a decimal number:20

When the user enters a decimal number, the program should print out the Gray code for the given decimal number as shown below:

Gray code for 20 -> 11110

and come back to the main menu. If the user selects (0), the program ends. Please restrict the user to enter inputs within the range [0,28-1=255] for the decimal number. For instance, if the user enters -1, 999, print out an error message and comes back to ask for correct inputs.

It is required to write a *modular* program. Please put the part of the code that outputs the Gray code in a new function called to\_Gray() inside the main.c file.

**Deliverables**

You will prepare and submit the program in one single zip file lab11\_{UWinID}.zip containing the following items:

1. The code files and executable file (main.exe in windows or main in unix/mac)

2. The result of the commands in the file results.png/jpg. Simply make a screenshot of the results.

3. [Optional and if necessary] A readme document in a txt file readme.txt. It explains how to build and run the program as well as any prerequisites that are needed. *Please note that if your program cannot be built and run on our computer systems, you will lose marks.*

Lab11\_hfani.zip

* (70%) main.c => Printing Gray code
* (05%) main.exe or main
* (10%) results.jpg/png
* (Optional) readme.txt

(10%) Modular Programming (using separate header and source files and functions)

(05%) Files Naming and Formats

*Please follow the naming convention as you lose marks otherwise.* Instead of UWinID, use your own UWindsor account name, e.g., mine is [hfani@uwindsor.ca](mailto:hfani@uwindsor.ca), so, lab11\_hfani.zip.

1. https://en.wikipedia.org/wiki/Frank\_Gray\_(researcher) [↑](#footnote-ref-1)