Assignment N^o1

Group 2

September 2, 2018

0.1 Excercise 1: Resolution and Range of a Fixed-Point Binary Representation

0.1.1 What is the Fixed-Point Binary Representation

A fixed-point number has an integer part and a fractional part separated by a decimal point with a fixed position, as shown below:

$$(IntegerPart).(FractionalPart)$$

The integer part is formed by n bits and the fractional part is formed by m bits.

$$(bit\#1 \quad bit\#2 \quad ... \quad bit\#n).(bit\#1 \quad bit\#2 \quad ... \quad bit\#m)$$

0.1.2 What is Resolution and Range

Resolution

The resolution of a number using the fixed point representation is the smallest unit that can be handled with it. Given a fixed-point number with m fractional bits, the resolution is 2^{-m}.

Range

The range is the difference between the biggest value that can be obtained with the fixed-point representation of a number with n bits in the integer part and with m bits in the fractional part, and the smallest number that can be represented.

0.1.3 Making Use of this Program

Input

Three arguments must be entered through Command Line, separated by one space:

1. 1 (indicating that the numeric representation of the binary number is signed) or 0 (indicating that the representation is unsigned).
2. n: A possitive integer (indicating the number of bits that correspond to the integer part of the number, which appears before the decimal point).

3. m: A possitive integer (indicating the number of bits that correspond to the fractional part of the number, which appears after the decimal point).

For example: "0 1 1".

Output

The result of this program is the resolution and range of the number that has n digits in the integer part and m digits in the fractional part.

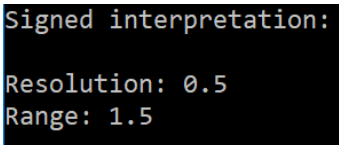

Figure 1: Output corresponding to the example input "0 1 1".

0.1.4 Testing the Program

0.2 Excercise 2: Simplification of a Maxterm Expression and its Corresponding Logical Circuit

Having the function in maxterms

$$f_1(A, B, C, D) = \prod (M_0, M_1, M_5, M_7, M_8, M_{10}, M_{14}, M_{15})$$

equivalent to

$$f_2(A, B, C, D) = \sum (m_2, m_3, m_4, m_6, m_9, m_{11}, m_{12}, m_{13})$$

using minterms, can be simplify by different ways and represented using logic gates.

0.2.1 Simplify: Boolean Algebra

Using the Boolean algebra propertie

$$(A + B).(A + \overline{B}) = A \ (1)$$

or

$$(AB) + (A\overline{B}) = A \ (2)$$

the function could be simplify using (1):

$$\begin{aligned} f_1(A,B,C,D) = {} & (A+B+C+D).(A+B+C+\overline{D}).(A+\overline{B}+C+\overline{D}).(A+\overline{B}+\overline{C}+\overline{D}). \\ & (\overline{A}+B+C+D).(\overline{A}+B+\overline{C}+D).(\overline{A}+\overline{B}+\overline{C}+D).(\overline{A}+\overline{B}+\overline{C}+\overline{D}) \\ = {} & (A+B+C).(A+\overline{B}+\overline{D}).(\overline{A}+B+D).(\overline{A}+\overline{B}+\overline{C}) \end{aligned}$$

Which in minterms would be, using (2):

$$\begin{aligned} f_2(A,B,C,D) = {} & (\overline{A}\,\overline{B}C\overline{D}) + (\overline{A}\,\overline{B}CD) + (\overline{A}B\overline{C}\,\overline{D}) + (\overline{A}BC\overline{D}) + \\ & (A\overline{B}\,\overline{C}D) + (A\overline{B}CD) + (AB\overline{C}\,\overline{D}) + (AB\overline{C}D) \\ = {} & (A\overline{B}D) + (\overline{A}B\overline{D}) + (\overline{A}\,\overline{B}C) + (AB\overline{C}) \end{aligned}$$

0.2.2 Simplify: Karnaugh Map

Karnaugh map is a easier way to simplify logic experesion when the functions are too complex or too large to handle, cause Karnaugh map gives a more representative view for a faster analisis for it to simplify.
If the simplification is done with minterms, the groups should be of 1, adding each group in case there is more than 1, and in each group the independent variables would be multiplied.

CD \ AB	00	01	11	10
00	0	1	1	0
01	0	0	1	1
11	1	0	0	1
10	1	1	0	0

Now grouping the colour groups we get that the function in minterms would be:

$$\begin{aligned} f_2(A,B,C,D) = &(A\overline{B}D)\ (Red) \\ &+(\overline{A}B\overline{D})\ (Blue) \\ &+(\overline{AB}C)\ (Orange) \\ &+(AB\overline{C})\ (Green) \end{aligned}$$

The same method could be done with maxterms; grouping O, multipling groups in case there is more than 1, and in each group the independent variables would be added.

0.2.3 Logic Circuit: AND, OR and NOT

Using the logic gates AND, OR and NOT the simplify version of the function could be represented in the figure below:

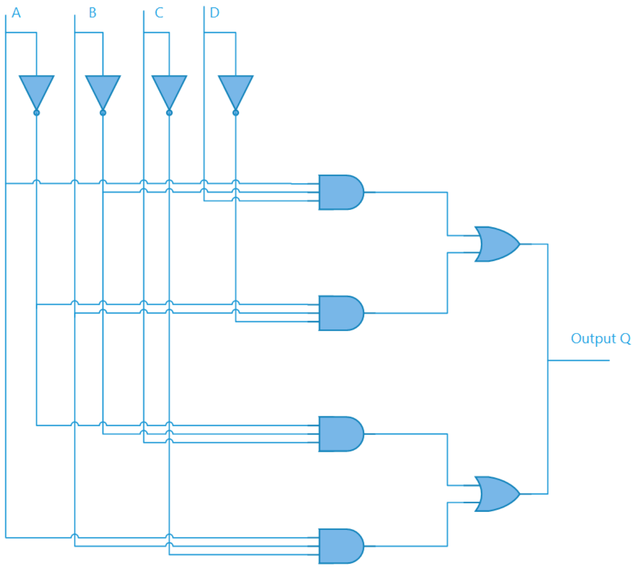

Figure 2: Logic circuit using AND, OR and NOT gates

0.2.4 Logic Circuit: NAND

All the gates could be equivalent to a combination of NAND or NOR gates. Therefore, the simplify function can be drawn as the next figure:

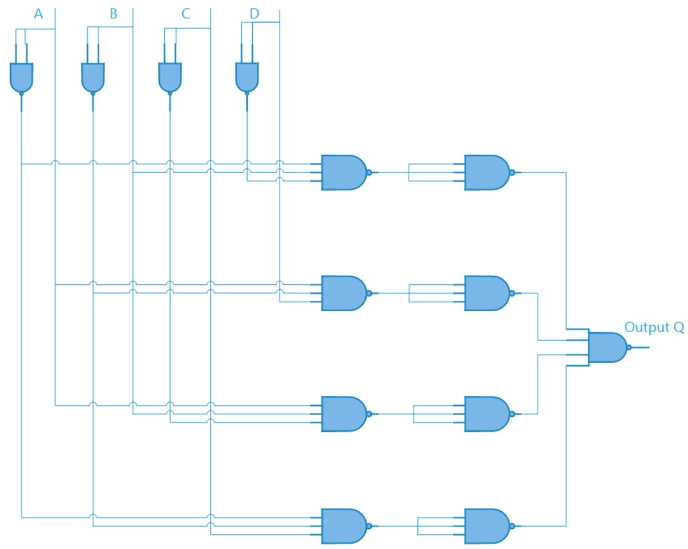

Figure 3: Logic circuit using NAND gates

0.3 Excercise 3: Four-Entry Encoder and Demux using Verilog

Implement the following modules in Verilog:

- 4 inputs ENCODER
- 4 outputs DEMUX

0.3.1 4-Input ENCODER

Description

An encoder is an Application-Specific Integrated Circuit (ASIC) that converts information. In this case it receives a signal from a 4-bit input and returns the position of the Most Significant Bit that is currently on.

Code Implementation

The Code Implementation of both the Module and its testbench can be found in their respective directories.

Module Tests

Results of the Testbench:

Input	Output	Value
0001	00	0
0010	01	1
0100	10	2
1000	11	3
0011	01	1
0101	10	2
1001	11	3
0110	10	2
1010	11	3
1100	11	3

Table 1.1.3 ENCODER Testbench Results

Conclusions

The module is working as expected, where it is taking only the Most Significant Bit as the value to be encoded.

0.3.2 4-output DEMUX

Description

A DEMUX is an ASIC which receives an input signal and a selector signal. The selector signal determines through which output port the input signal is sent.

Code Implementation

The Code implementation for the DEMUX can be found in its corresponding folder.

Module Tests

Input	Selector	Out_0	Out_1	Out_2	Out_3
1	0	1	0	0	0
1	1	0	1	0	0
1	2	0	0	1	0
1	3	0	0	0	1

Table 1.2.3 DEMUX Testbench Results

Conclusions

The module is works as expected.