Digital Circuits Theory - Laboratory						
Academic year	Laboratory exercises on	Mode of studies	Field of studies	Supervisor	Group	Section
2024/2025	Wednesday 11:45 – 13:15	SSI	Informatics	KP	1	1

Report from Exercise No 11

Performed on: 20.11.2024

Exercise Topic: Implementing logic functions with MUX and DMUX

Performed by:

Piotr Copek
Zuzanna Micorek

Introduction

In this report, we present the design and implementation of various combinational circuits. These tasks involved the use of Karnaugh maps and the application of multiplexers and demultiplexers to implement the desired functions. We employed different configurations of multiplexers [8-to-1, 4-to-1, and 2-to-1] and a combination of a 1-to-4 demultiplexer with a 4-to-1 multiplexer. Each circuit was built by grouping and simplifying Boolean expressions, ensuring efficient and correct logic implementation. This report summarizes our methodology and the solutions developed, providing insights into combinational circuit design.

Task 1.c

Implementation of function Z_2 with use of 8-to-1 Multiplexer and gates:

$$Z_2 = \sum(4, 5, 6, 7, 9, 10, 11, 12, 13, 14)_{x_3x_2x_1x_0}$$

Solution

- We began by filling out a Karnaugh map for the function Z_2 with single address input x_3.
- Each column of the K-map corresponds to an input combination for the 8-to-1 multiplexer. We numbered these columns to indicate the input lines for the multiplexer.

x_3 \ $x_2x_1x_0$	000	001	010	011	100	101	110	111
1	1	0	1	0	1	0	1	0
0	0	1	0	1	0	1	0	1
	D_0	D_1	D_3	D_2	D_6	D_7	D_5	D_4

Z_2

Figure 1 - Karnaugh map for task 1.c with description of each column.

- We formed the equations for each column ($D_0 \rightarrow D_7$).

$$D_0 = 0$$
$$D_1 = D_2 = D_3 = x_3$$
$$D_4 = D_5 = D_6 = 1 d_7 = \overline{x_3}$$

- At the end we connected the multiplexer inputs accordingly.

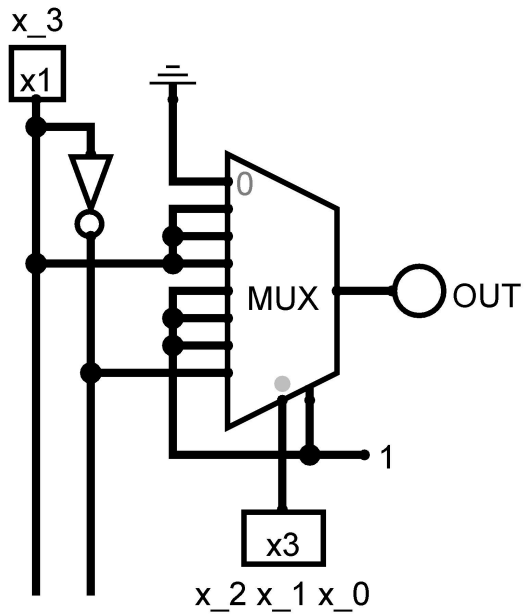

Figure 2 - Implementation for task 1.c with output function Z_2 denoted as OUT. On Laboratories we only had access to 16-bit multiplexer so we had to ground one unused MSB address bit.

Summary

In this task, we designed a combinational logic circuit to implement the function Z_2 using an 8-to-1 multiplexer. We started by filling out a Karnaugh map, created groups for minimization, and connected the multiplexer based on the derived expressions. This task provided insights into using multiplexers for implementing complex logic functions.

Task 1.d

Implementation of function Z_2 with use of 4-to-1 Multiplexer and gates:

$$Z_2 = \sum (4, 5, 6, 7, 9, 10, 11, 12, 13, 14)_{x_3 x_2 x_1 x_0}$$

Solution

- Karnaugh Map Initialization: We began by filling out a Karnaugh map for the function Z_2 with two address inputs, x_1 and x_0.
- Each column in this K-map represents an input combination for the 4-to-1 multiplexer. We numbered these columns accordingly.

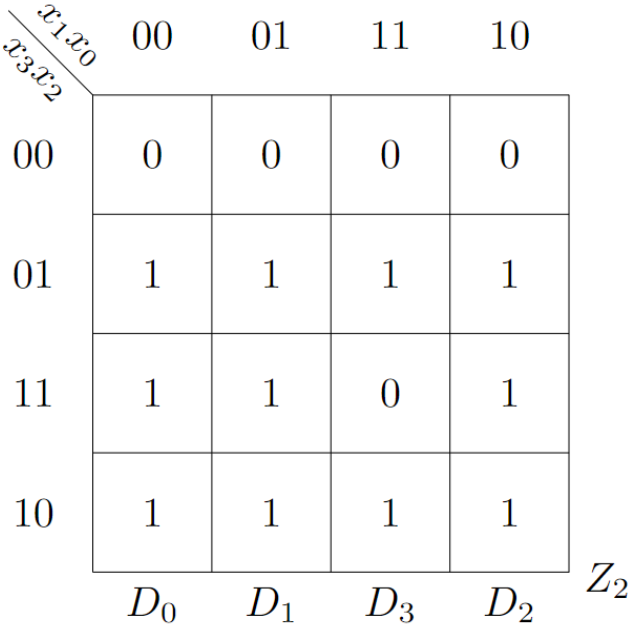

Figure 3 - Karnaugh map for task 1.d with description of each column.

- We formed the equations for each column ($D_0 \rightarrow D_3$).

$$D_0 = x_2$$
$$D_1 = D_2 = \overline{\overline{x_3} \cdot \overline{x_2}}$$
$$D_3 = x_3 \oplus x_2 = \overline{\overline{\overline{x_3} \cdot x_2} \cdot \overline{x_3 \cdot \overline{x_2}}}$$

- At the end we connected the multiplexer inputs accordingly with use of NAND gates.

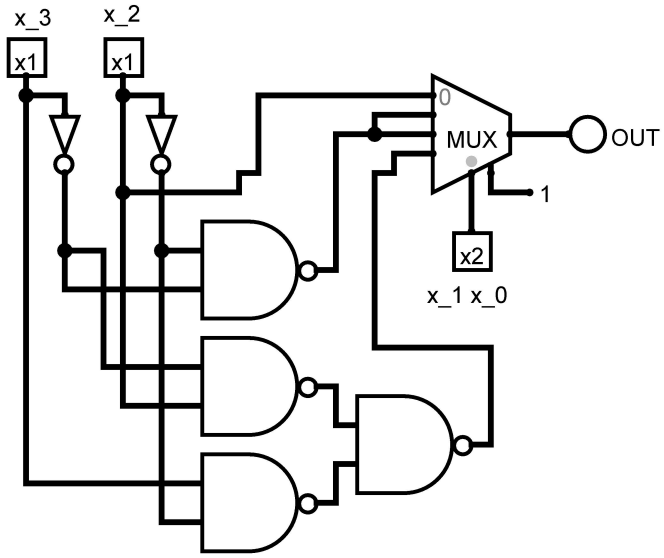

Figure 4 - Implementation for task 1.d with output function Z_2 denoted as OUT. On Laboratories we only had access to 16-bit multiplexer so we had to ground two unused MSB address bits.

Summary

In this task, we designed a combinational logic circuit to implement the function Z_2 using an 8-to-1 multiplexer and gates. We started by filling out a Karnaugh map, created groups for minimization, and connected the multiplexer based on the derived expressions with use of NAND gates.

Task 1.e

Implementation of function Z_2 with use of 2-to-1 Multiplexer and gates:

$$Z_2 = \sum(4, 5, 6, 7, 9, 10, 11, 12, 13, 14)_{x_3 x_2 x_1 x_0}$$

Solution

- We filled out a Karnaugh map for the function Z_2 using three address inputs x_3, x_2 and x_1.
- Each column in the K-map corresponds to an input combination for the 2-to-1 multiplexer, which we numbered accordingly.

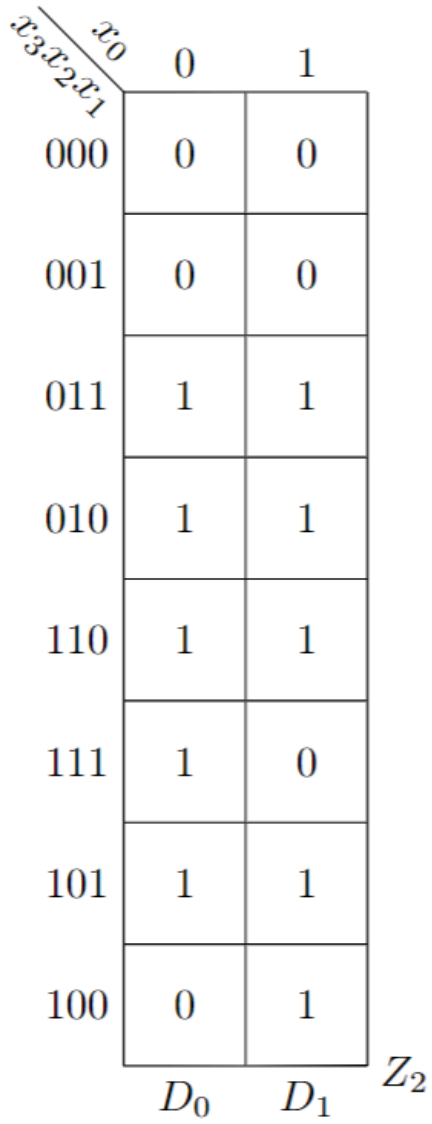

Figure 5 - Karnaugh map for task 1.e with description of each column.

- We formed the equations for each column ($D_0 \rightarrow D_1$).

$$D_0 = \overline{\overline{x_2} \cdot \overline{x_3 \cdot x_1}}$$

$$D_1 = \overline{\overline{\overline{x_3} \cdot x_2} \cdot \overline{x_2 \cdot \overline{x_1}} \cdot \overline{x_3 \cdot \overline{x_2}}}$$

- At the end we connected the multiplexer inputs accordingly with use of NAND gates.

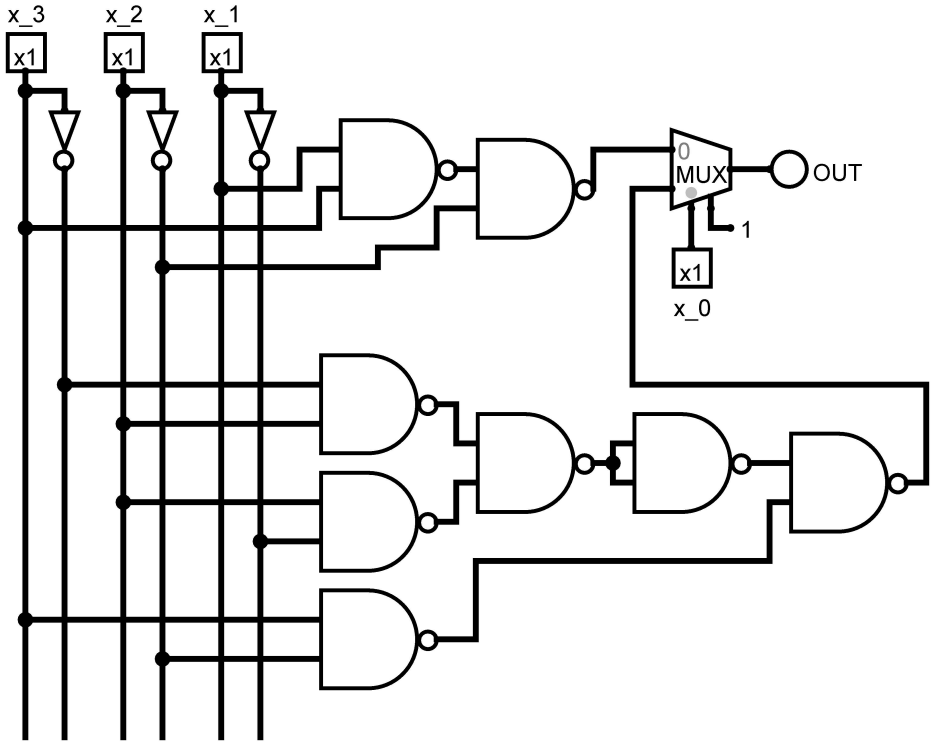

Figure 6 - Implementation for task 1.e with output function Z_2 denoted as OUT. On Laboratories we only had access to 16-bit multiplexer so we had to ground three unused MSB address bits.

Summary

In this task, we implemented the function Z_2 using a 2-to-1 multiplexer and gates. We filled the Karnaugh map and derived the necessary Boolean expressions and connected the multiplexer inputs. Task showed that implementation of complex boolean expressions does not necessarily require complex and big multiplexer.

Task 1.f

Implementation of function Z_2 with use of 1-to-4 Demultiplexer and 4-to-1 Multiplexer structure:

$$Z_2 = \sum(4, 5, 6, 7, 9, 10, 11, 12, 13, 14)_{x_3x_2x_1x_0}$$

Solution

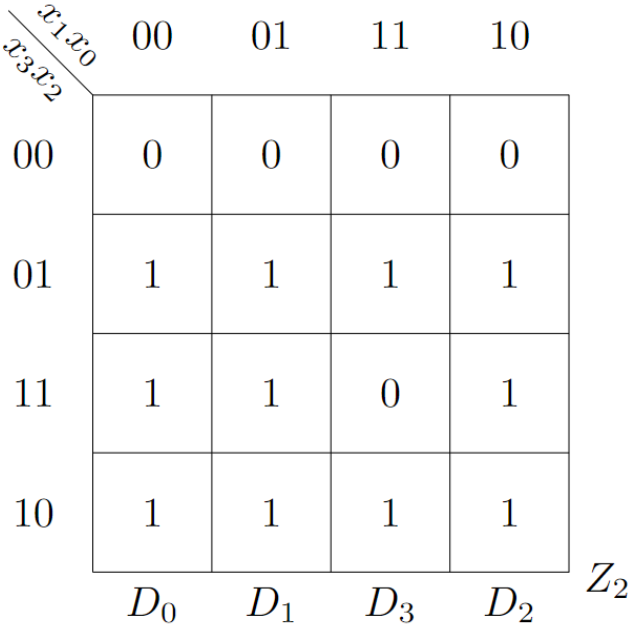

Figure 7 - Karnaugh map for task 1.f with description of each column.

- We formed the equations for each column ($D_0 \rightarrow D_3$).

$$D_0 = \begin{cases} \sum(1,3)_{x_3x_2} \\ \prod(0,2)_{x_3x_2} \end{cases}$$

$$D_1 = D_2 = \begin{cases} \sum(1,2,3)_{x_3x_2} \\ \prod(0)_{x_3x_2} \end{cases}$$

$$D_0 = \begin{cases} \sum(1,2)_{x_3x_2} \\ \prod(0,3)_{x_3x_2} \end{cases}$$

- At the end we connected the multiplexer inputs accordingly with use of NOR gates.

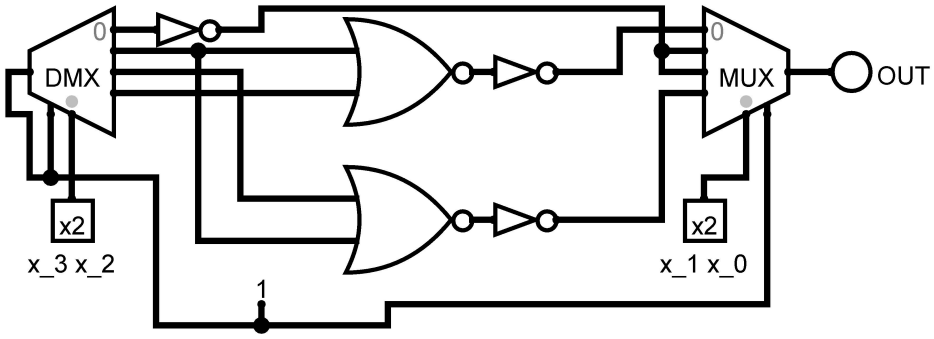

Figure 8 - Implementation for task 1.f with output function Z_2 denoted as OUT. On Laboratories we only had access to 16-bit multiplexer so we had to ground two unused MSB address bits.

Summary

In this task, we implemented the function Z_2 using a 2-to-1 multiplexer and gates. We filled the Karnaugh map and derived the necessary Boolean expressions and connected the multiplexer inputs. Task showed that implementation of complex boolean expressions does not necessarily require complex and big multiplexer.

Final Conclusions and Observations

In these tasks, we designed and implemented various combinational logic circuits to achieve specific digital functions. We started by creating Karnaugh maps and forming groups for simplification. We used different types of multiplexers and a demultiplexer to implement the function Z_2. These exercises allowed us to gain practical experience in combinational logic design. By working with 8-to-1, 4-to-1, and 2-to-1 multiplexers, as well as a combination of a demultiplexer and multiplexer, we understood the versatility and application of these components in digital circuits.