|  |  |  |
| --- | --- | --- |
| G:\nsu-logo.png  **North South University**  Department of Electrical & Computer Engineering    **LAB REPORT**  **Computer Organization and Architecture Lab**  Experiment Number: **Lab -** **#01**   |  | | --- | | Experiment Name: **Design of a 2-bit Logic unit** |     Experiment Date: 30-10-2021  Report Submission Date: 02-11-2021  Section: **02** | |
| Student Name: **Asfaria Islam Chowdhury** | Score |
| Student ID: **1931741642** |  |
| Remarks: |

**Objectives**

Here, a two bit logic unit is being used, which is part of an arithmetic logic unit. The ALU here performs microoperations of AND, OR, XOR, and NOT.

Microoperations are instructions that are used in the BUS and control systems, on either individual bits or on a word portion that the register obtains from the memory. Examples of microoperations include making one’s complement on a group of bits or clearing a group of bits from a register.

In the experiment, it is demonstrated that via selection bits into multiplexers, the ALU determines which of the four microoperations should take place, i.e. more than one microoperation out of the four listed cannot take place simultaneously.

A and B are two-bit inputs, so the four outputs A AND B, A OR B, A XOR B, and A NOT are 2-bit.

**List of Equipments**

1. Trainer board
2. IC 7404,7408,7432,7486, 74F153
3. Wires for connection

**Theory**

The following are first performed (to later connect to multiplexer):

1. A0 AND B0 , A1 AND B1
2. A0 OR B0 , A1 OR B1
3. A0 XOR B0 , A1 XOR B1
4. A0 NOT, A1 NOT

The four output lines from the 0th bit are connected as inputs for a 4x1 MUX0, and the four output lines from the 1st bit are connected as inputs for a 4x1 MUX1. The outputs of the multiplexers are labeled as F0 and F1 respectively.

The two multiplexers must have the same 2 selection bits, S0 and S1. The values for these selection bits are provided by the ALU in order to choose which microoperation to perform. The table drawn on the next page lists down which combination of selection bits perform which microoperation.

|  |  |  |
| --- | --- | --- |
| **S0** | **S1** | **Microoperation** |
| 0 | 0 | AND |
| 0 | 1 | OR |
| 1 | 0 | XOR |
| 0 | 1 | NOT |

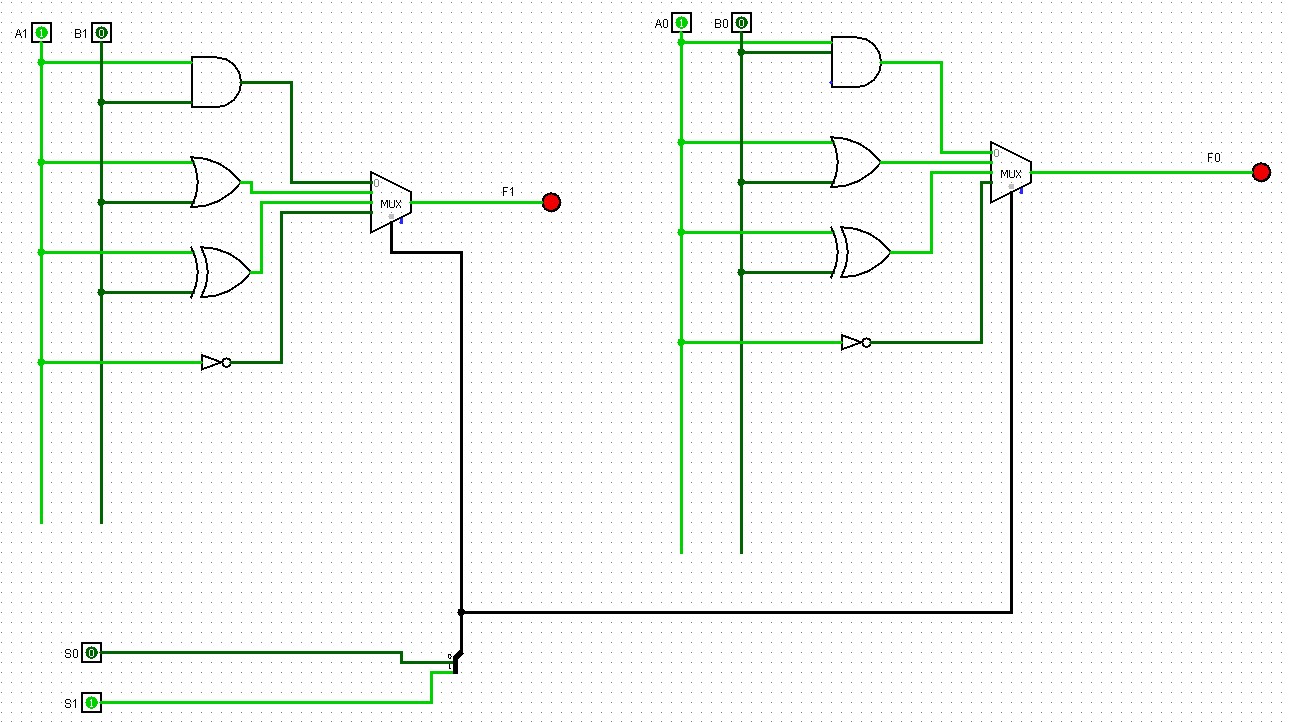
An example would be to choose to see OR microoperation when A = 11 and B = 00. First, all of the four logical functions are performed before even connecting to MUX and before even choosing the desired microoperation:

1. A0 AND B0 = 1, A1 AND B1 = 1
2. A0 OR B0 = 1, A1 OR B1 = 1
3. A0 XOR B0 = 0, A1 XOR B1 = 0
4. A0 NOT = 0, A1 NOT = 0

In order to perform OR microoperation, the selection bits according to the table above must be 01. The output of the multiplexers is the output of A OR B.

**Circuit Diagram**

Here, the simulation is for the example written above, in Theory section, for A=11 and B=00.



**Truth Table**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** | **AND1** | **AND0** | **OR1** | **OR0** | **XOR1** | **XOR0** | **NOT1** | **NOT0** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

**Discussion**