Lab 2

Dr. A. Kavianpour

1-Lab2 Assignment

A-Write a VHDL code for 4-1 MUX. This MUX has 4 inputs, one output, and two select inputs Call the file **mux41.vhd**

B-Write test bench for 4-1 MUX. Call the file **tb-mux41.vhd**. Test your code with input D3D2D1D0=1011 and selects lines 01 and 11.

C- Write a VHDL code for 16-1 MUX using only several 4-1 MUX similar to part (A). Use component and portmap instructions. Call this file **mux161.vhd**

D-Write test bench for 16-1 MUX. Call the file **tb-mux161.vhd**. Test your code for inputs D15D14...D0=7DAB in hexadecimal and select lines 0011, 1010, and 1110

Inputs (HEX)	Select Lines
7DAB	0011=3
	1010=10
	1110=14

2-Lab submission: Two items

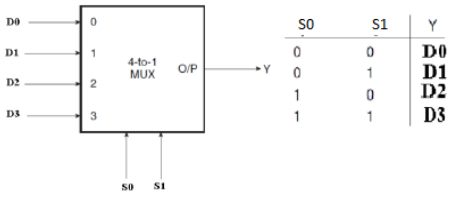
(a)-Lab report in doc or pdf copy and paste your codes (mux41.vhd, mux161.vhd, tb_mux41, tb_mux161.vhd) and screenshots of all tests for both 4-1 and 16-1 MUX. Mention if your design is working.

(b)-Vhdl files (mux41.vhd, mux161.vhd, tb_mux41, tb_mux161.vhd)

Introduction and background

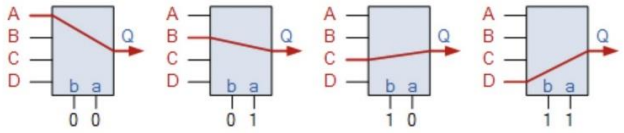
4-1 Multiplexer

A multiplexer or data selector, is a combinational circuit having multiple input line, select lines and one output line. Further, some multiplexers integrated circuits are having ENABLE input, which has to be activated to perform the multiplexer operation. In multiplexer depending upon select lines the binary data present on inputs is passed to the output line. If there are $\boldsymbol{n}$ select lines, then the maximum input lines are $\mathbf{2}^{\boldsymbol{n}}$ and the multiplexer is referred to as a $2^n to 1$ multiplexer or $2^n \times 1$ multiplexer. Figure below show the block presentation and truth table of 4-to-1 multiplexer.



S0	S1	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3

A more graphical representation is as follows. Note that A,B,C and D, can be m-bit inputs, but b and a (select lines) should be 1-bit inputs:



16- 1 Multiplexer

In this section, we explore the design of a 16-1 MUX using 4-1 MUX. Based on what mentioned in last section, we need 4 select lines for 16 (2^4) input lines. A 16x1 MUX can be implemented using 5 blocks of 4 × 1 MUXs. 4 of these multiplexers can be used as first stage to MUX 4 inputs each with two least significant bits of select lines (S0 and S1), resulting in 4 intermediate outputs, which, then can be MUXed again using a 4 × 1 MUX. The implementation of 16x1 MUX using 4x1 MUXs is shown in figure below: