

Experiment 06 Multiplier and ALU	EEL2020 Digital Design Department of Electrical Engineering IIT Jodhpur	
---------------------------------------------------	--------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------

Experiment 06

Objectives

- (i) Design, simulate and implement a 2-bit multiplier using half adders and AND gates (ii) Design and simulate an Arithmetic Logic Unit (ALU) that performs eight operations selected using operation codes

(i) Two-bit Multiplier using HA and AND gates

A two-bit multiplier can be implemented as follows:

$$\begin{array}{r}
 A_1 A_0 \\
 \times B_1 B_0 \\
 \hline
 A_0 B_1 A_0 B_0 \\
 + A_1 B_1 A_1 B_0 \times \\
 \hline
 C_2 A_1 B_1 + C_1 A_0 B_1 + A_1 B_0 A_0 B_0
 \end{array}$$

A one-bit multiplication can be implemented using a two-input AND gate.

The 2-bit multiplication output be a 4-bit product ($P_3 P_2 P_1 P_0$) would involve

- $P_0 = A_0 B_0$: Simple product of LSBs (Can be implemented with one AND gate) -
- P_1 = Addition of $A_0 B_1$ and $A_1 B_0$ (can be implemented with a half adder)
- P_2 = Addition of $A_1 B_1$ with carry of the previous HA (can be implemented with another half adder)
- P_3 = Carry out of the second HA, if any.

The logic circuit representing the above operation is shown in Figure 1.

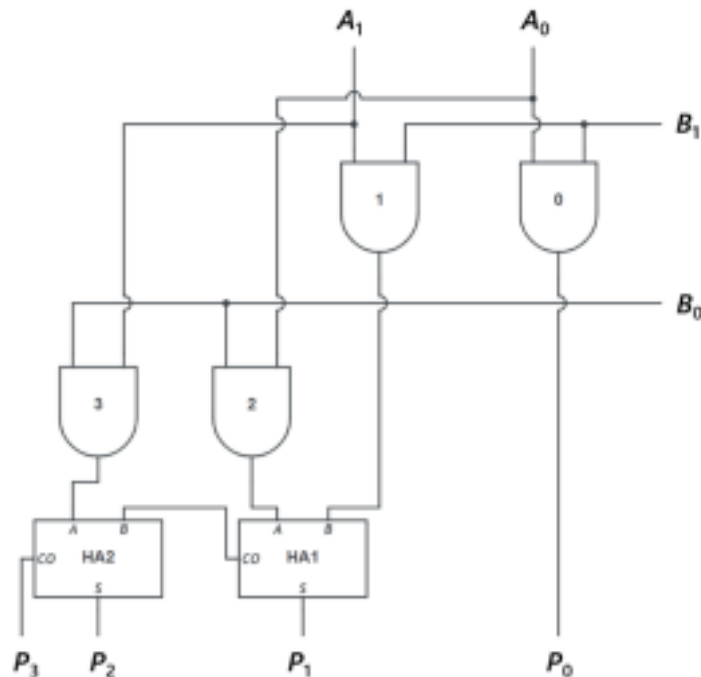



Figure 1 Logic diagram of a 2-bit multiplier circuit using half adders

Experiment 06 Multiplier and ALU	EEL2020 Digital Design Department of Electrical Engineering IIT Jodhpur	
---------------------------------------------------	--------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------

Procedure

Step 1: Write a Verilog module to represent the logic circuit shown in Figure 1. Reuse the half adder module created in Experiment 01.

Step 2: Write a testbench to simulate the results for at least eight inputs combinations. Verify the timing diagram with the truth table and **save the screenshot for the report.**

Step 3: Run the RTL Analysis. Save the **Elaborated Design** (insert in your report) and verify if it matches with your logic design. (If not, explain how a change in your Verilog module may have generated a different elaborated design.)

Step 4: Use the four slide switches on the RPI module as inputs (eg. Input A_1A_0 on RPI Module Switches (H, G) and input B_1B_0 on RPI Module Switches (F, E)). To display the 4-bit product $P_3P_2P_1P_0$, use the four LEDs (LD3 – LD0) on the PYNQ-Z2 board. **Update the same on the RPI and PYNQ Z2 XDC files (and include the updates in your report).**

Step 5: Generate bitstream and program the device. Verify the working of all combinations of inputs. **Record a video showing all input/output combinations.**


(ii) Arithmetic Logic Unit (ALU)

An Arithmetic Logic Unit (ALU) is a fundamental component of a CPU (Central Processing Unit) that performs arithmetic and logical operations on operands. It is responsible for executing operations like addition, subtraction, AND, OR, etc., depending on the instruction provided by the control unit. The

instruction from the control unit can be provided in the form of an operation code. Assuming a very simple example of an Operation Table shown in Table 01, design and implement the corresponding ALU that performs these eight operations based on the corresponding 3-bit operation code.

Table 1 Operation Codes and their corresponding arithmetic or logic operations

Opcode			Operation
0	0	0	Reset
0	0	1	A + B
0	1	0	A – B
0	1	1	A AND B
1	0	0	A OR B
1	0	1	A XOR B
1	1	0	A XNOR B
1	1	1	Preset

Experiment 06 Multiplier and ALU	EEL2020 Digital Design Department of Electrical Engineering IIT Jodhpur	
---------------------------------------------------	--------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------

Create a testbench showing all eight operations on at least two different inputs. Insert the timing diagram and verify the operations.

Run the RLT Analysis. Analyse and discuss the elaborated design created for your ALU.

Online Reference

R. Chouhan, *EEL2020 Digital Design Lab – Multiplier and ALU* (<https://youtu.be/dtRQtfHa0hk>)