

# ECE 201 Lab - Design Project 1

## A 4x4 Combinational Multiplier

### Objectives

To practice the combinational design process through the design of a 4-bit multiplier.

### Introduction

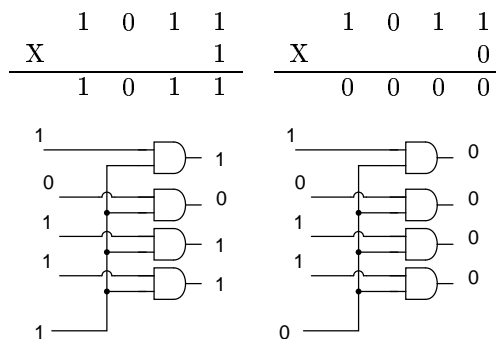
This lab will be unlike previous labs in two very significant ways. First, we're going to design circuits that are too complex to fit on our breadboard with the parts in the lab kit, so this lab will be done *entirely* with the simulator. Second, though this lab writeup will provide some insights into a possible design of a multiplier, no circuit diagrams will be provided – you'll do the design yourself (woohoo. OK, now this time with enthusiasm, this is what you're here for: Woohoo!).

The circuit you will be designing is a 4-bit multiplier, i.e. a circuit with two 4-bit numbers as inputs, and the 8-bit product of these two numbers as the output.

A truth table for this circuit would have 8 inputs, 8 outputs, and 256 lines. You would need eight 8 variable K-maps to directly produce boolean equations. So there's no way around it, we're going to have to think some about this one. The following section describes the multiplication process in detail, in terms of steps we've already figured out how to do. After that, it's up to you.

### Background

The first thing you need to know about multiplication is that the AND operation will multiply two bits together. It works just like decimal multiplication,  $1*1=1$ , and  $0*0=0*1=1*0=0$ . AND even has the same symbol as decimal multiplication. So, if you need to multiply a number by a single bit, you can just use a bunch of AND gates. Consider the example below:





				$A_3$	$A_2$	$A_1$	$A_0$
$X$				$B_3$	$B_2$	$B_1$	$B_0$
				$P_{30}$	$P_{20}$	$P_{10}$	$P_{00}$
+			$P_{31}$	$P_{21}$	$P_{11}$	$P_{01}$	
		$S_{14}$	$S_{13}$	$S_{12}$	$S_{11}$	$S_{10}$	$P_{00}$
+			$P_{32}$	$P_{22}$	$P_{12}$	$P_{02}$	
	$S_{24}$	$S_{23}$	$S_{22}$	$S_{21}$	$S_{20}$	$S_{11}$	$P_{00}$
+		$P_{33}$	$P_{23}$	$P_{13}$	$P_{03}$		
	$S_{34}$	$S_{33}$	$S_{32}$	$S_{31}$	$S_{30}$	$S_{20}$	$S_{10}$
						$S_{10}$	$P_{00}$

where  $S_{ij}$  represents the  $j$ th output of the  $i$ th adder. Note that at each stage you need a 4-bit adder (which would have five outputs), and keep in mind that you have already built four bit adder macros in an earlier lab.

## Procedure

Design it, simulate it, turn it in.

Bring an electronic copy of your design to lab with you, and be prepared to demonstrate the operation of your functional multiplier (Add appropriate switches and lights so you can input two 4-bit numbers and see your 8-bit output). Make sure your work is your own! Turn in a printout of your final schematic, plus printouts of any macros you used in your designs as well. Also turn in a brief report describing your circuit design. In your report, consider if and how well this circuit design would scale up to multiplying bigger numbers. Also, answer the following questions in your report:

- What would be the strengths and weaknesses of a large multiplier built in this fashion?
- What would the propagation delay be for the circuit in your simulation, assuming a 1 ns delay for each gate?

## A note for those who are interested

The multiplier you just built is pretty close to the kind used in high-end arithmetic circuits, a kind of multiplier known as an *array multiplier*. Real array multiplier pass the carry through the adders to the last one, rather than using ripple or look ahead adders to add the partial sums. A special look ahead adder is designed for the last stage to speed the whole thing up. Multipliers built in this way are about the highest performance possible. However, many multipliers are not built in this fashion. A typical chip will use a much smaller, slower, but more complicated multiplier design based on registers (which you'll hear about in the next couple of weeks).