

# ECE 201 Lab - Binary Arithmetic

## Objectives

The student should demonstrate knowledge of simple binary arithmetic, and the mechanics of its use. Each student is required to design, simulate, build, and test a 2-bit full adder.

## Requirements

A complete, simulated circuit diagram and a discussion of the full adder should be included in the report. A simulated diagram of a two-bit full adder should also be provided.

## Background

By now, you have become familiar with how to do arithmetic with binary numbers. Let's examine how to generate boolean equations (and therefore how to build a circuit) that describe this process.

To begin with, consider the problem of adding two single bit numbers. This can result in a two-bit answer ( $1 + 1 = 10$ ). The resulting circuit will therefore need to have two inputs and two outputs. The truth table for this circuit is shown below:

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

The two output functions are labeled "C" and "S". The "S" is simple, this stands for "sum". The "C" stands for "carry", since, if we were adding numbers that were bigger than a single bit, the MSB of the result would become the carry for the next bit. These two functions can be written as two MSOP equations:

$$C = AB$$

$$S = A'B + AB'$$

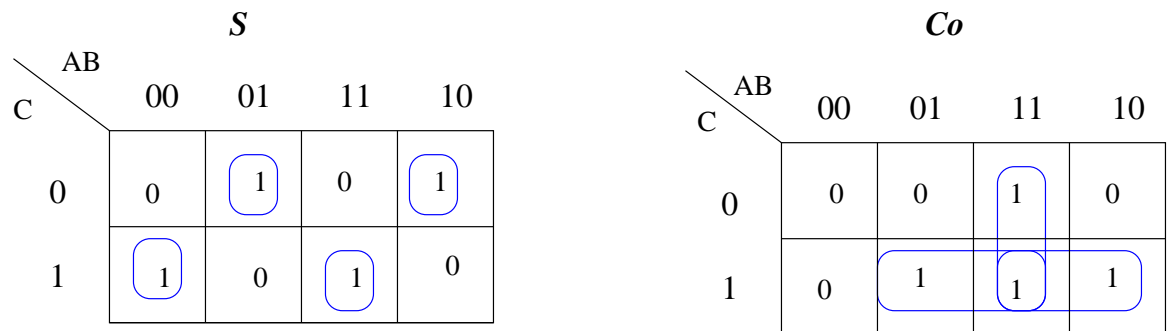
A device that implements these two functions is known as a *half adder*. This adder is referred to as "half", because it only solves half the general problem of adding numbers with more than one bit. Let's take a look at an example of what happens when we add two (unsigned) 8 bit numbers:

(Carry)	(0)	(1)	(1)	(1)	(0)	(0)	(0)	
A=	1	0	1	1	0	0	0	1
B=	0	0	1	0	1	1	0	0
	1	1	1	0	0	1	0	1

Note that except for the right most column, we are actually adding *three* bits: a bit from each of the 2 numbers and a carry bit from the bits immediately to the right. Note also that each addition produces 2 bits - the result bit and the carry bit. Now, let's make a truth table for this addition process. The truth table will have three variables, 1 bit from each of the numbers A and B, and a carry in bit,  $C_{in}$ , which represents the carry from the previous position. The two outputs are the sum bit and the carry out bit,  $C_{out}$ , which will be used in the next position.

$C_{in}$	A	B	$C_{out}$	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

We'll use Karnaugh maps to simplify the two functions in the table above into MSOP form:

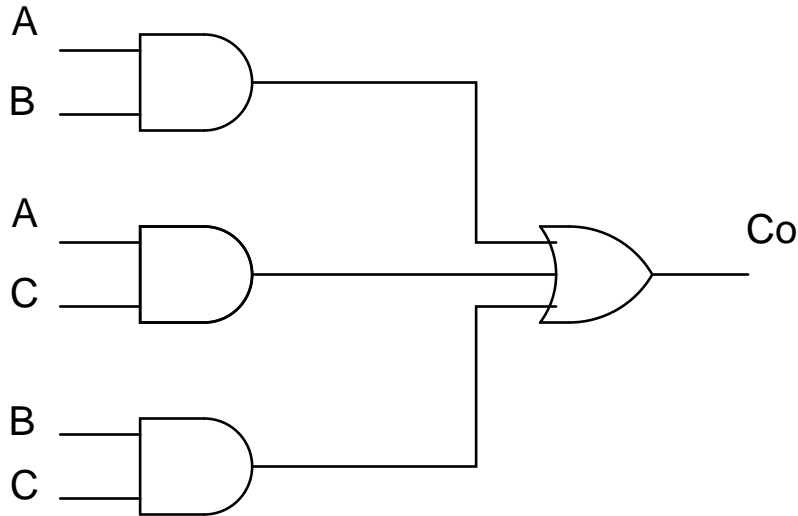


Using the groupings shown in the map above, we get the following MSOP functions:

$$S = A'B'C + AB'C' + A'BC' + ABC$$

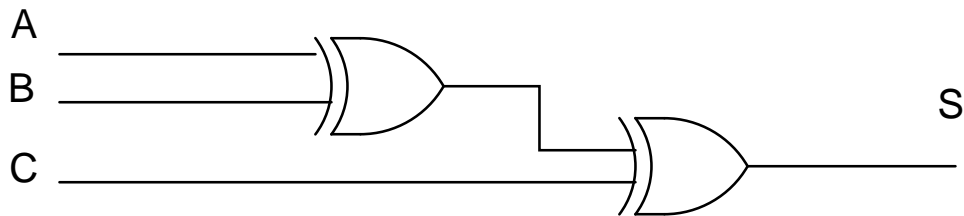
$$C_{out} = AB + AC + BC$$

We can implement the function for  $C_{out}$  in a straightforward manner, as shown below:



S is a bit more complex if we implement it in MSOP form. However, if we examine this function a bit more closely, we will see the now familiar checkerboard pattern in the K-map, and notice that S is only equal to 1 when an *odd* number of the input variables are 1 in the truth table. S can therefore be easily implemented with an EXOR function, as shown below:

$$S = A \oplus B \oplus C$$

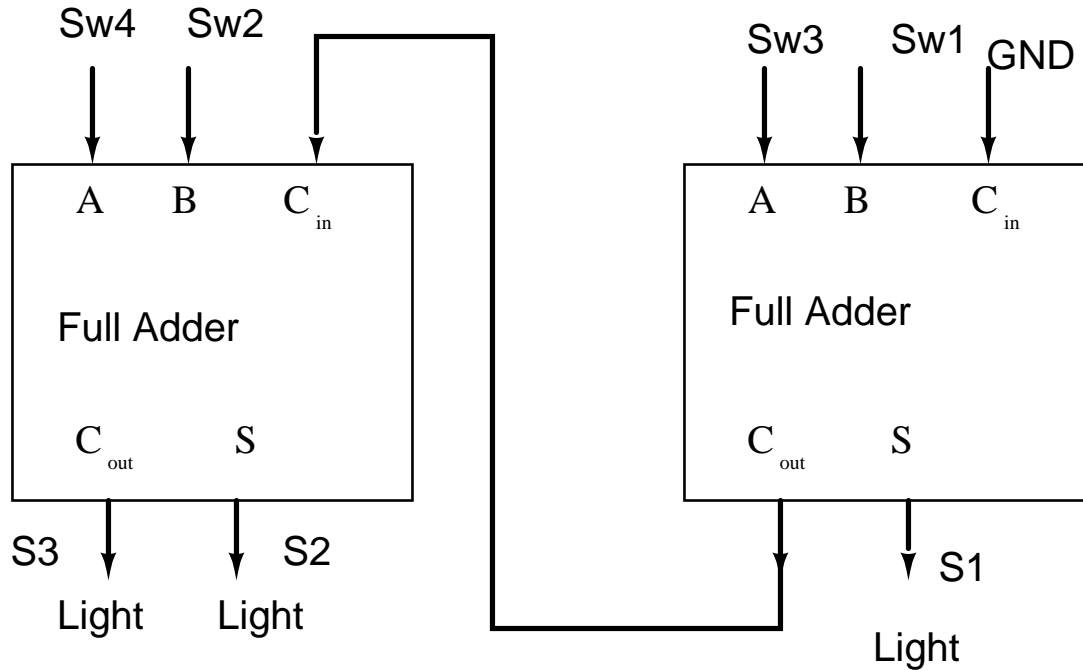


These two circuits together are called a *full adder*.

## Procedure

You are going to build a device which will add 2 unsigned 2 bit numbers. Use a pair of switches for each input value. The results will be displayed on LEDs. You will need to build two copies of the full adder. The carry input to the right most adder will be tied to GND. The carry in of the left adder will be tied to the carry out of the right adder. What happens to the carry out of the left adder?

In block diagram form:



You will need the following chips to build this circuit:

1 - 7486

2 - 7408

2 - 7432

**A note about simulation:** In this simulation, you will build some simple circuits with gates, then use multiple copies of those circuits to build a larger, more complicated device. This is typical of how system design is done. We use this same kind of procedure with Computer Aided Design (CAD) tools, like your digital simulator. Ideally, you would like to enter the circuit for a full adder once, test and debug it, then turn it into a part we can use over and over again. Then you could simply use two of those parts and draw the connections between them to build your two bit adder. This is called a *hierarchical* design. In Digital Works, hierarchy is achieved by using *macros*. You used a macro in the previous lab for the 7447 driver chip. This week, you will want to create your own macro for the full adder circuit. See the online Digital Works “Getting Started” guide for details on how. Once your macro is created, you will create *another* schematic that uses two copies of the macro, and adds the switches and lights. Your resulting schematic should look like the block diagram of the circuit shown above.

Turn in copies of the schematic for you full-adder macro as well as your final circuit in lab.

*HINT:* It will probably be easier if you use one 7408 and one 7432 for each carry generator rather than using one chip for parts of both full adders. This way

the two carry circuits can have identical pin assignments and also be physically separate to help avoid confusion. This also makes it possible for you to label the pins correctly inside your full adder macro.