Adding 2 Bits

Understanding the Problem:

Consider two bits 0, 1 and add them together in all possible combinations.

a ->	0		0		1		1
b ->0	0	0	1	0	0	1	1
s ->	0		1		1		0

In decimal: 2 3

4_1 8

1

The one in the units column of the answer is the half sum, and the 1 that is brought over is called the carry out (subscript on the 4).
In binary there's always a half sum (s) and a carry out, which is in blue above. In each column you add three things, the carry in (which was the carry out from the column before), and the two numbers.

Putting the Info into the Form of 2 Truth Tables

a	b	s
0	0	0
0	1	1
1	0	1
1	1	0

a	b	carry
0	0	0
0	1	0
1	0	0

1	1	1

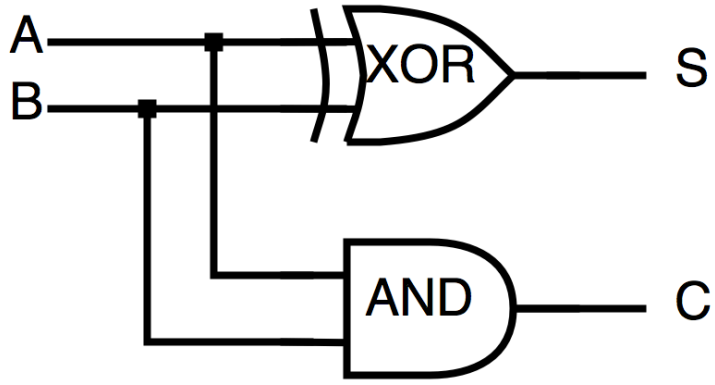
These truth tables are XOR (for the half sum) and AND (for the carry).

So adding a to b can be done with logic gates:

sum = a XOR b

carry = a.b

So What Does the Circuit Look Like?



Where S is the half-sum and C is the carry-out.

He showed us a program called Multimedia Logic which we can use to construct logical circuits and test them.

Looking More Closely at the "Carry" Process

Decimal Example:

0	7	9	6
0_1	2_1	4_1	8_0
1	0	4	4

So each time we're adding 3 things, the two digits and the carry-in. Have to do this if we want to do multi-digit addition.

From this we get two outputs, the sum and the carry-out.

The carry-out becomes the carry-in for the next addition.

carry-out <- |a| <- carry-in

This component (the attempted drawing) is called a Full Adder.
We can combine full-adders to make multi-digit adders, connecting the carry-out from each to the carry-in from the next.
(There were a lot of diagrams around here, so I'm missing a lot)
The carry ripples through the circuit and so the circuit is called a ripple-carry adder.
There are alternatives to a ripple-carry adder that are faster (each digit addition has to wait for the one before it to finish), but this is the simplest.

For the labs: He wants us to keep careful notes on each lab, and a log of the labs you've done.