Adding Bits With Logic Gates (cont.)

Example:

		1	1	1	0	1	1	0	1	
		0_1	1_1	1_0	0_1	1_1	1_0	1_0	0_0	
c_{out}->	1	0 |	1	0	1	1	0	1	1	<-sum

Truth Table for a Full Adder

We want to create a logical circuit to implement a full adder. Lets start with the truth table:

C_{in}	a	b	sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Later we will see a mechanism for creating equations and circuits from truth tables.

For now, we are going to create a circuit by intuition and pattern-matching.

C_{in}	a	b	sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1

1	1	1	1	1

Consider the two blocks above.

In the first block $c_{in} = 0$, in the second $c_{in} = 1$.

In the first block, sum = aXORb and $c_{out} = a.b$, which is the same as the half adder.

In the second block, sum = aXNORb and $c_{out} = a+b$.

Rewriting

If $c_{in} = 0$:

sum = aXORb, $c_{out} = a.b$

If $c_{in} = 1$:

sum = aXNORb, $c_{out} = a+b$

The following circuit implements this: (get picture)

a and b connected to each circuit listed above appropriately.

Those circuits connected to c_{in} by 2 multiplexors.

Multiplexor:

One input to an AND can control the gate. If it's 0, the gate outputs a 0. If it's 1 the gate outputs the other input to it.

By putting an OR after two ANDs, where one input to one AND is c_{in} and one input to the other AND is c_{in}', the value of c_{in} controls whether the result is taken from the first AND or the second AND.