




Building an Computer Adder

[Building an Adder]

- No matter how complex the circuit, or how complex the task being solved, at the base level, computer circuits are made up of three basic components.
- These basic components or gates are

- AND 
- OR 
- NOT 

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline \end{array}$$

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline \end{array}$$

1

Building an Adder

- Let's put this information in a table.

A	0	0	1	1
+				
B	0	1	0	1
Sum	0	1	1	10

- If we rotate this table placing the A, B and Sum at the top we get what looks like a standard truth table.

A	B	Sum
0	0	0
0	1	1
1	0	1
1	1	10

[Building an Adder]

- A computer will treat each column of digits in our binary addition as one operation.

$$\begin{array}{r}
 1010 \\
 + 111 \\
 \hline
 \end{array}$$

1

- Essentially the computer uses the circuit designed to implement the table below to solve each column.

A	B	S
0	0	0
0	1	1
1	0	1
1	1	(1)0

↑ Ignore the 1 which carries to next column for the moment!

Building an Adder

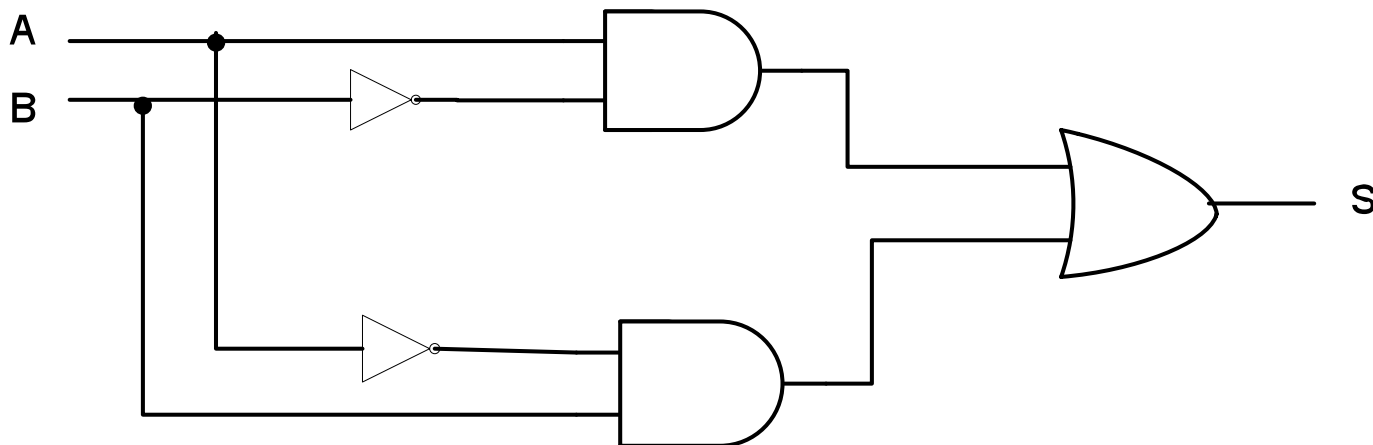
- Comparing this truth table to the ones for AND, OR and NOT, it is clear this is none of the above.

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

- This truth table is close to being a table for OR, but the final set of values doesn't match.

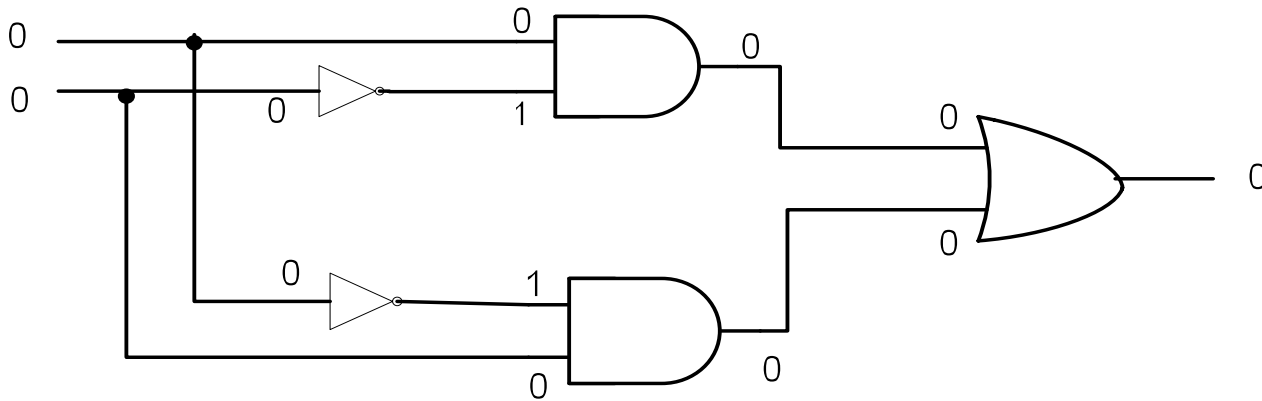
Building an Adder

- Using basic logic gates computer architects designed the following circuit to match the truth table for adding two binary digits.



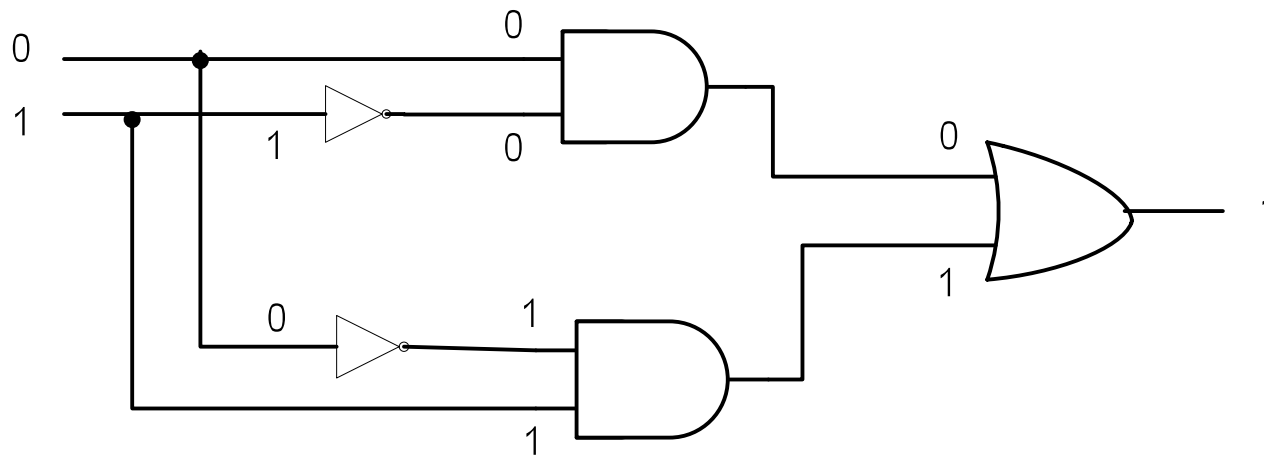
[Building an Adder]

- If $A = 0$ and $B = 0$, then $\text{Sum} = 0$



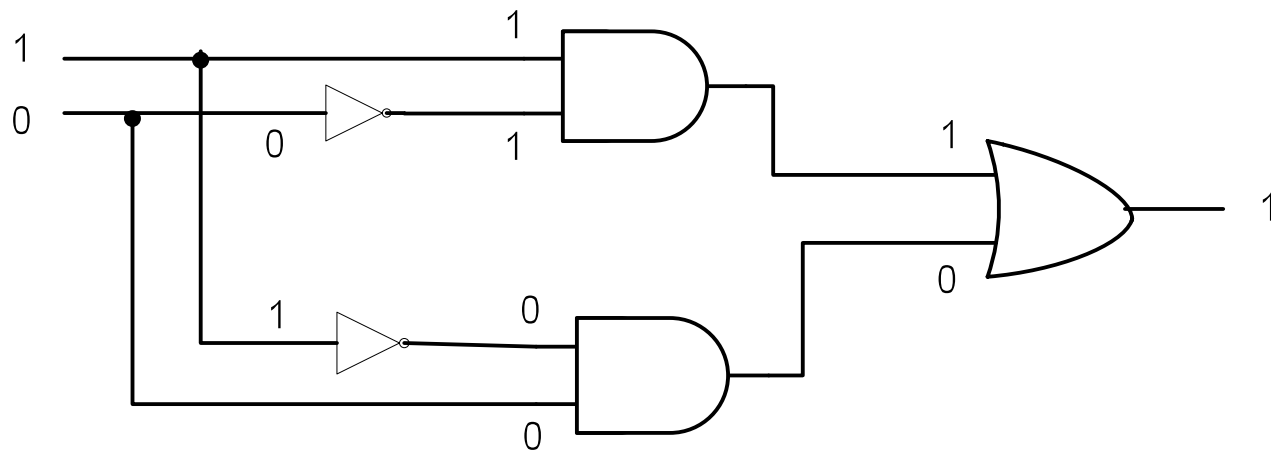
[Building an Adder]

- If $A = 0$ and $B = 1$, then $\text{Sum} = 1$



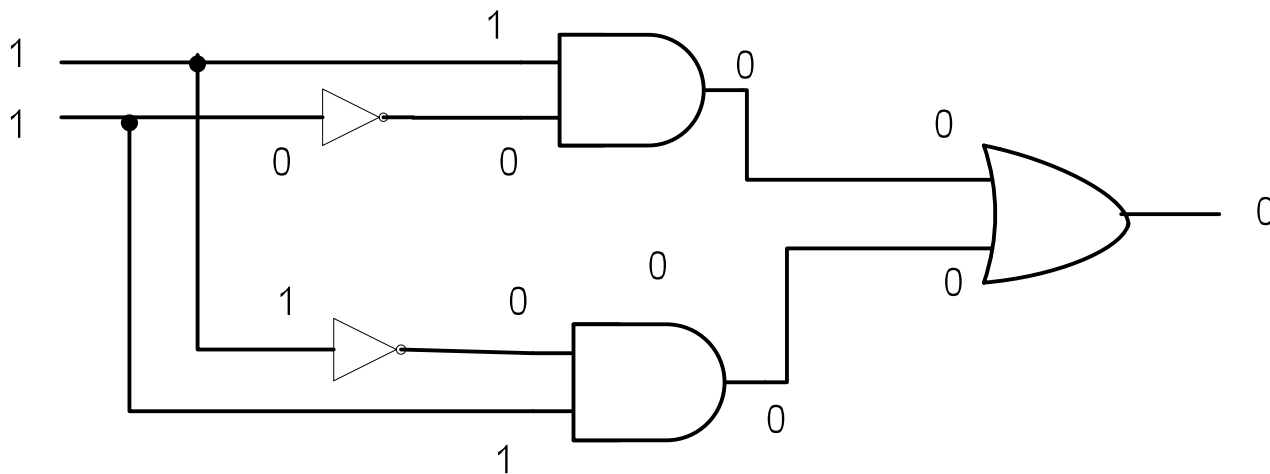
[Building an Adder]

- If $A = 1$ and $B = 0$, then $\text{Sum} = 1$

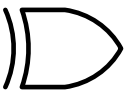


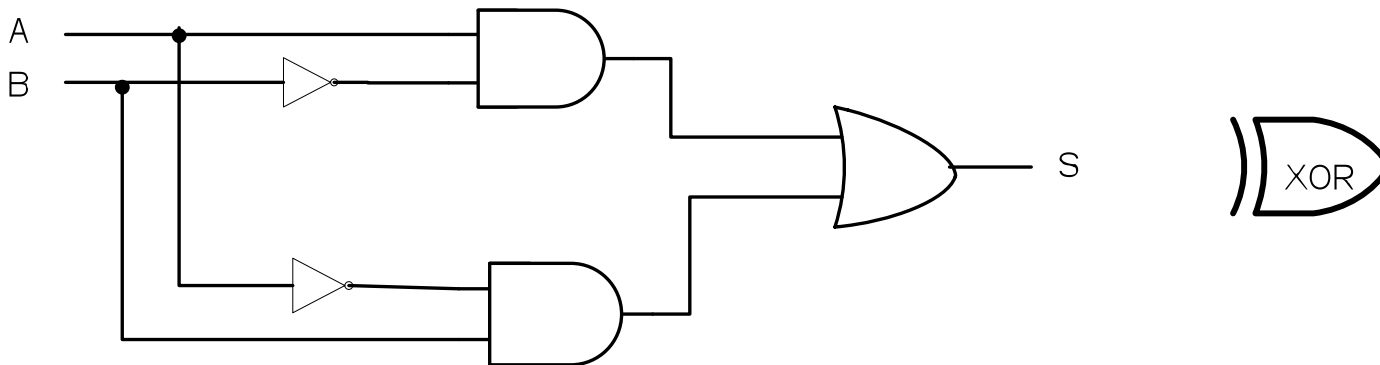
[Building an Adder]

- If $A = 1$ and $B = 1$, then $\text{Sum} = 0$.



Building an Adder

- This circuit has proved so important that it is given its own name.
- This gate is called an eXclusive OR and is given the symbol:  (in a logic statement \oplus)



[Building an Adder]

- Complete the binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline \end{array}$$

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline \end{array}$$

1

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline 01 \end{array}$$

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1 \\ 1010 \\ + 111 \\ \hline 01 \end{array}$$

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline 001 \end{array}$$

[Building an Adder]

- Examine the following binary addition problem:

$$\begin{array}{r} 1010 \\ + 111 \\ \hline 10001 \end{array}$$

[Building an Adder]

- When reviewing the example, the addition is not quite as simple as $A + B = \text{Sum}$.
- In many cases a “carry” impacts our addition.

$$\begin{array}{r} 1010 \\ + 111 \\ \hline 1011 \end{array}$$

Building an Adder

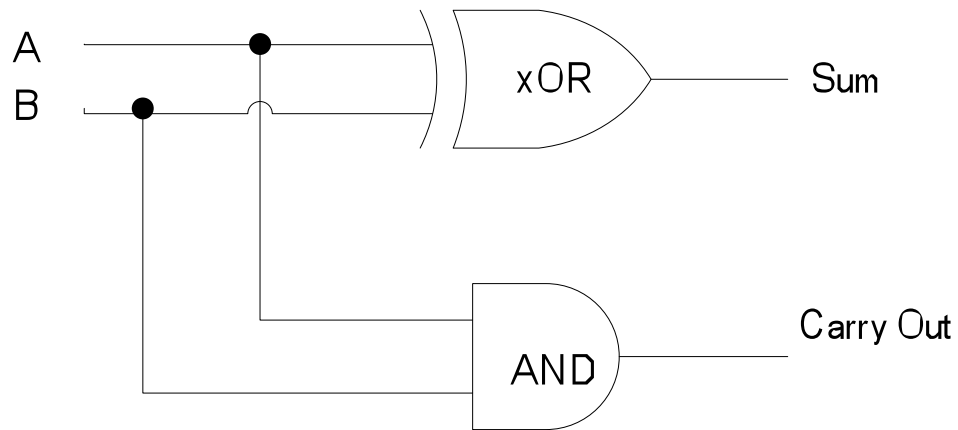
- Looking at our example one column at a time we notice the following:

A	B	Sum	Carry Out
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- Sum = $A \oplus B$ (A XOR B)
- Carry Out = AB (A AND B)

Building an Adder

- To accurately reflect single digit binary addition our circuit is:

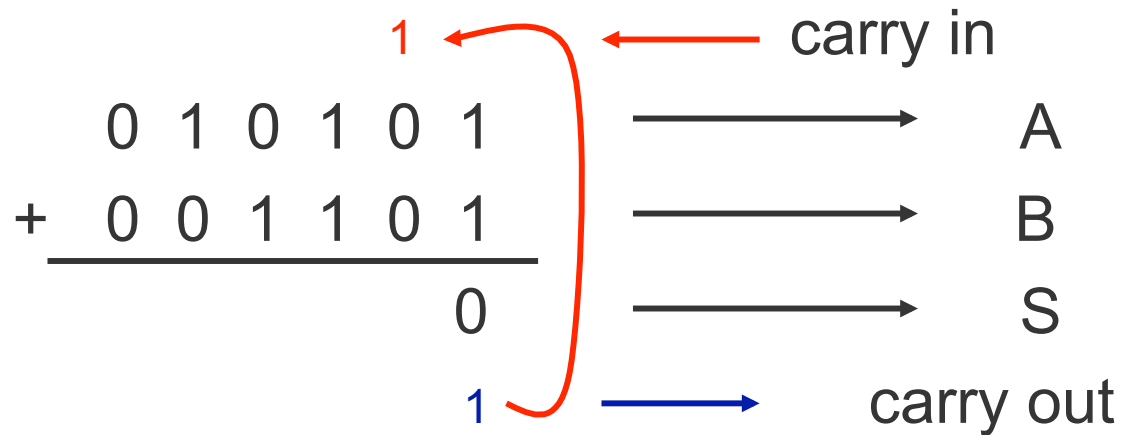


- Checking this circuit, it matches the truth table.

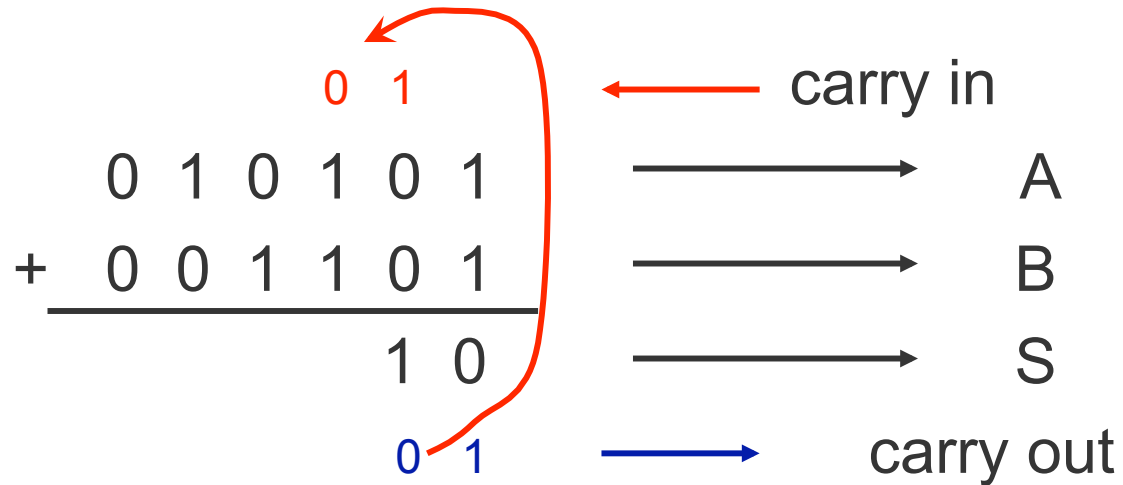
[Building an Adder]

- This is still not quite accurate.
- Going back to our binary addition problem, we find that in reality we are not adding two bits, and getting a two bit answer.
- Rather, our Carry Out from one column become the Carry In to the next column.

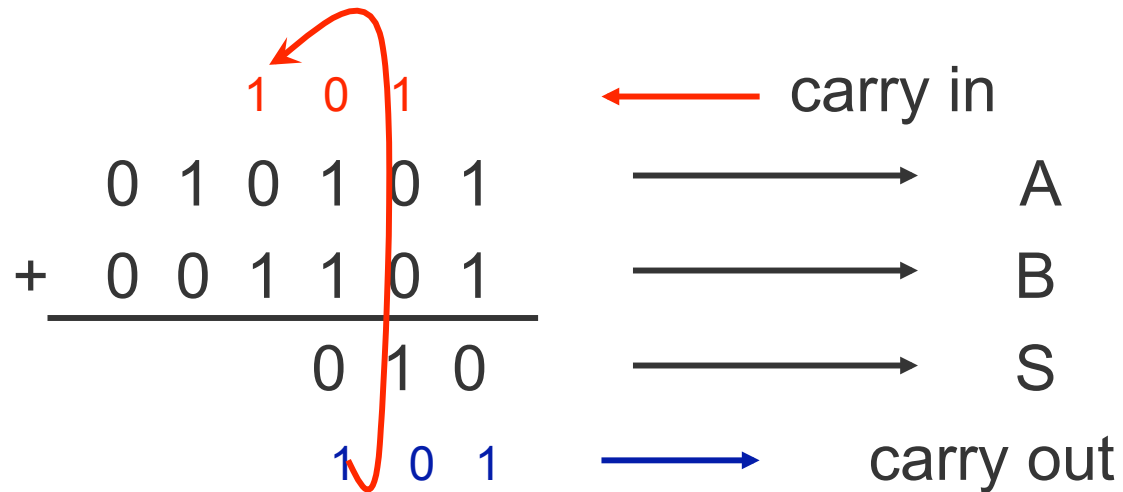
[Building an Adder]



[Building an Adder]



[Building an Adder]



[Building an Adder]

$$\begin{array}{r} 1\ 1\ 1\ 0\ 1 \\ 0\ 1\ 0\ 1\ 0\ 1 \\ +\ 0\ 0\ 1\ 1\ 0\ 1 \\ \hline 1\ 0\ 0\ 0\ 1\ 0 \\ 0\ 1\ 1\ 1\ 0\ 1 \end{array}$$

carry in

A

B

S

carry out

[Building an Adder]

- Adding two 1-bit numbers together turns out to really require that we add 3-bits together (including our Carry In) and producing an answer that is 2-bits (including our carry out).
- A complete or “full” adder 3-bits, C_{in} , A , B and produces a Sum and a Carry Out.

Building an Adder

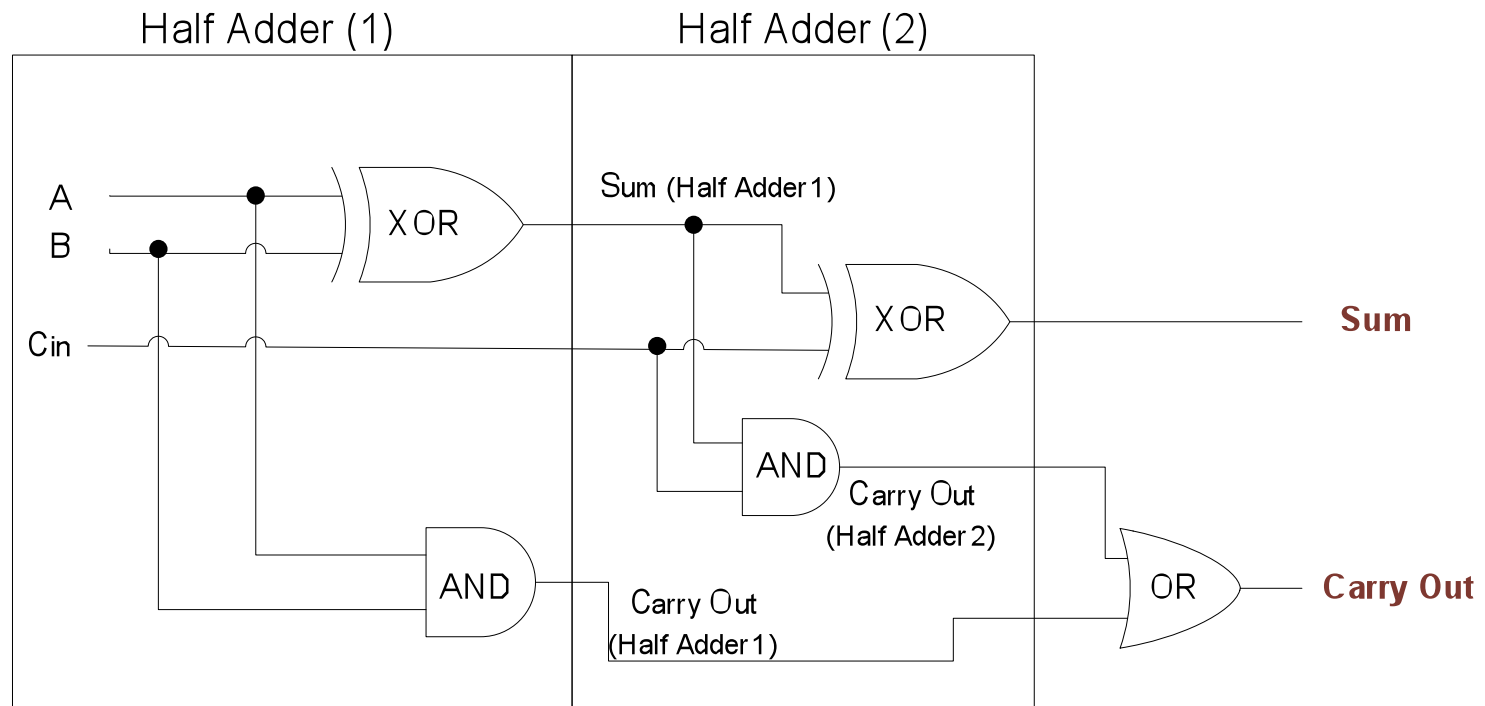
- The truth table that represents all the possible outcomes of these bits would look like this: This table can be represented by the following logic statements:

- **Sum = (Cin ⊕ (A ⊕ B))**
- **Cout = (A ⊕ B)(Cin) + AB**

Cin	A	B	Sum	Cout
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

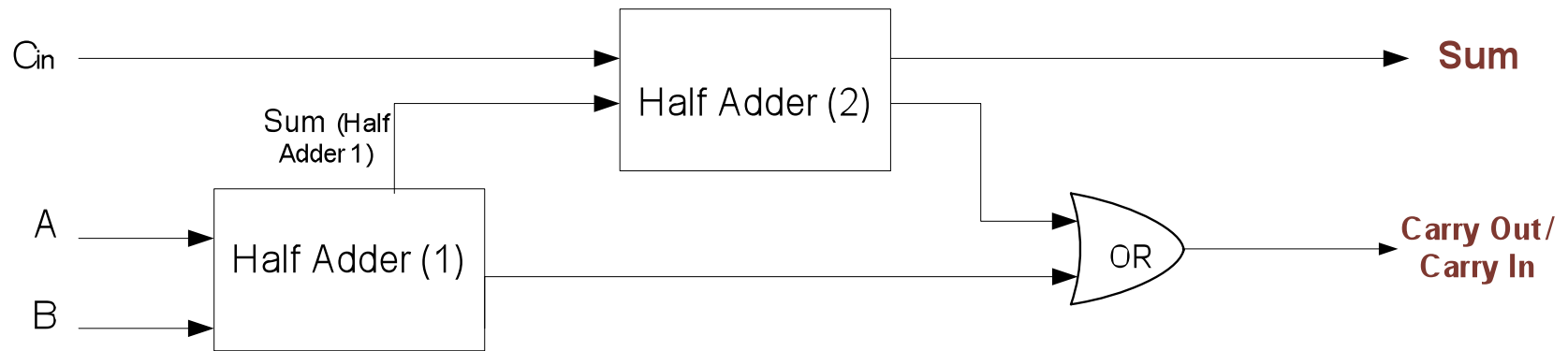
[Building an Adder]

- The circuit that represents this truth table, is called a Full Adder.



[Building an Adder]

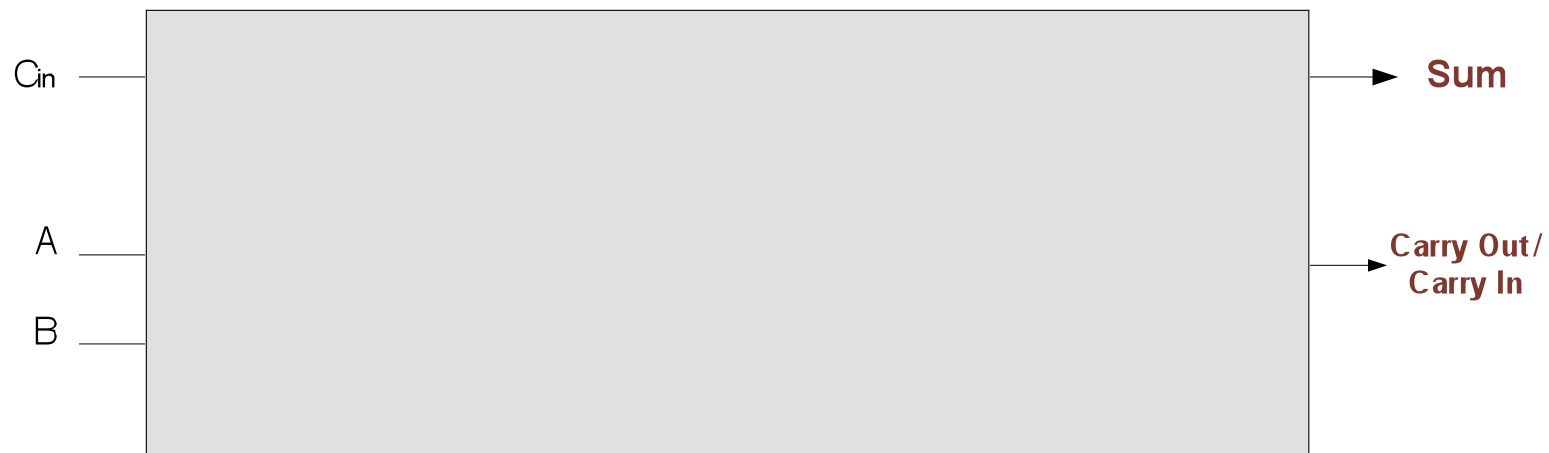
- Another way to look at this is:



[Building an Adder]

- A Full Adder then looks like this:

Full Binary Adder



[Building an Adder]

- Going back to our binary addition, we find that a Full Adder, adds one column at a time.

$$\begin{array}{r} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \\ + \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \\ \hline \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \\ \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \begin{array}{|c|} \hline 0 \\ \hline \end{array} \begin{array}{|c|} \hline 1 \\ \hline \end{array} \end{array}$$

[Building an Adder]

- To perform real binary addition, a series of binary adders are linked together.
- The number of linked adders is determined by the manufacturer and reflects the finite-length a particular computer can handle.
- This is usually, 8-bits, 16-bits, 32-bits or more.

Building an Adder

- A series of Full Adders would look like this:

