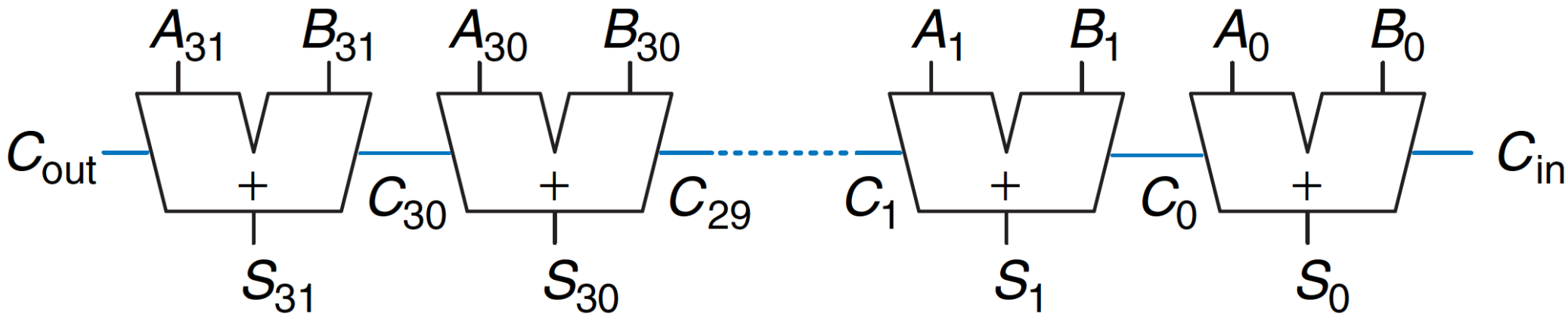
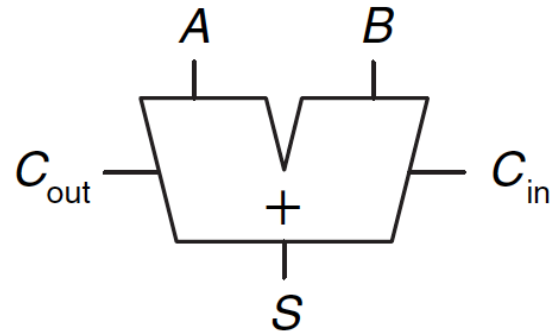
Timing of Ripple-Carry Adder

Ripple Carry Adder



Full Adder

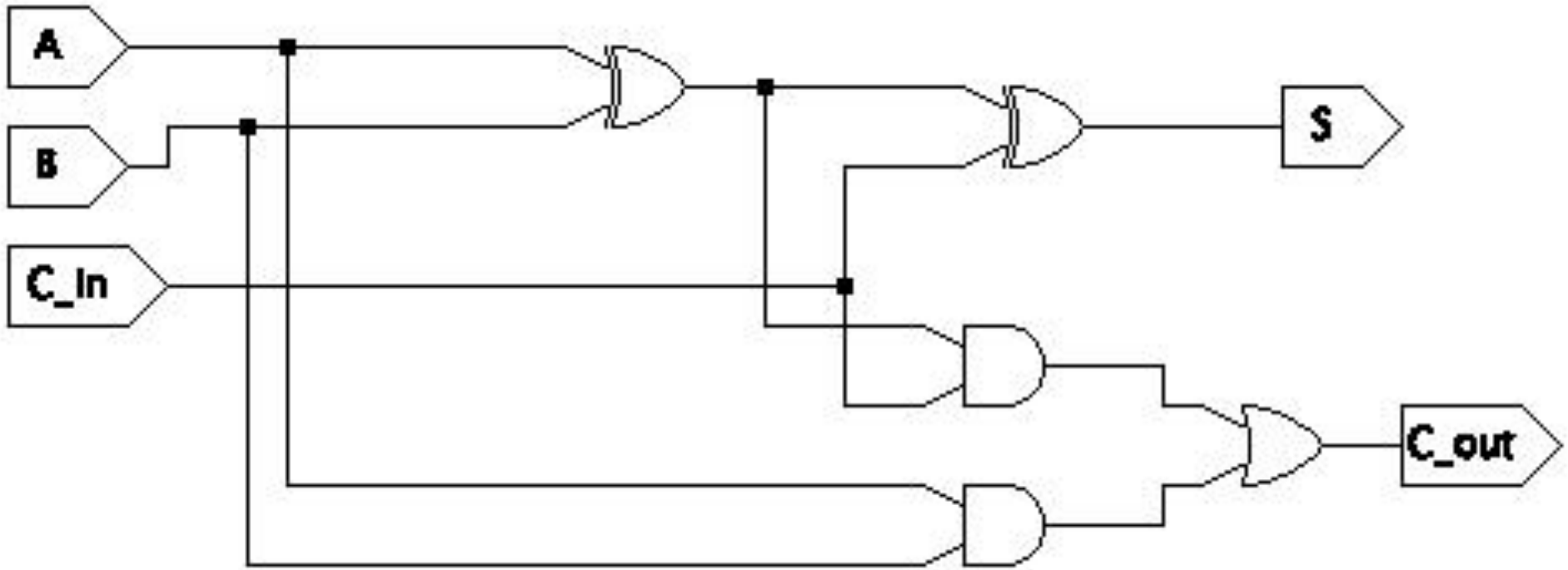


C_{in}	A	B	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = AB + AC_{in} + BC_{in}$$

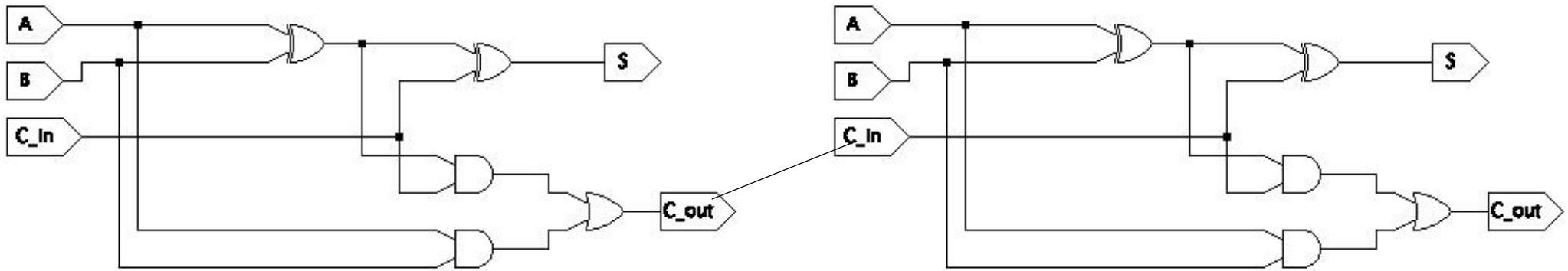
Full Adder



Critical path of full adder

Sum (S) computed after two gate delays

C_{out} requires three gate delays

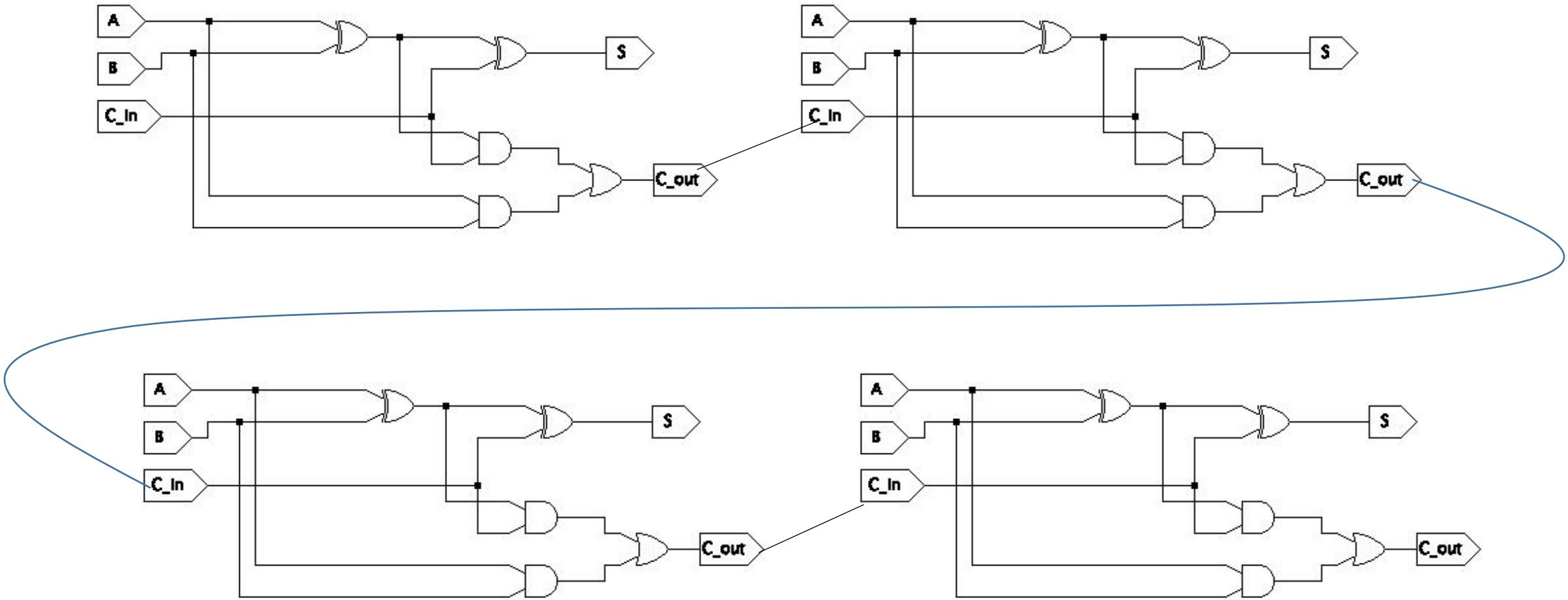

A
B
C_In
S
C_out
A
B
C_In
S
C_out

Working in Parallel

Critical path of 2-bit ripple-carry adder is just five gates, not six!

Very important point about working with hardware

Everything without a dependency happens in parallel


A
B
C_In
S
C_out
A
B
C_In
S
C_out
A
B
C_In
S
C_out
A
B
C_In
S
C_out

Timing of ripple-carry adder

First adder requires three gate delays

Remaining adders require just two gate delays

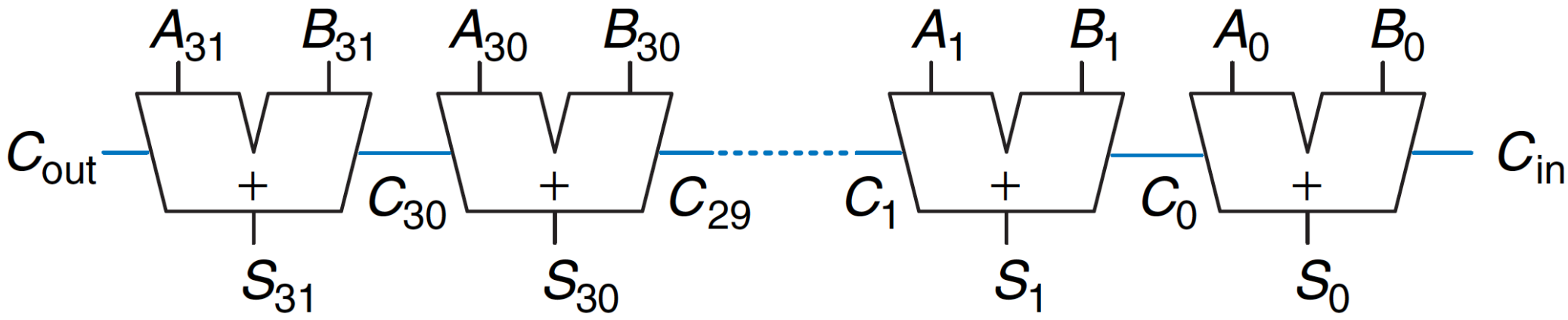
For N-bit adder:

$$3t + (N-1)*2t =$$

$$2Nt + t$$

where t is the gate delay

Ripple Carry Adder



Propagation delay

Recall that we concern ourselves only with *slowest possible* route through circuit

This includes assuming worst-case inputs

If adder never produced carries (e.g., 0 + 0), it would be finished much sooner than our formula indicates

However, if we knew that, we would not need to bother adding the numbers

Before checking output of adder, need to know that it will be finished for all possible inputs