Addition

Steven R. Bagley

Binary Addition

- Build a circuit to add two binary numbers together
- First let’s recap how addition works
- Think about the process we do…
- Since we need to build that process in hardware

Addition

```
  1 0 2 4
  4 0 9 6 +
-----------
```

Standard decimal addition

Addition

```
  1  0  2  4
  4  0  9  6  +
--------------
           0
```

Standard decimal addition

Add 4 and 8 get 12 or 2 and carry 1

Addition

```
  1  0  2  4
  4  0  9  6  +
 --------------
           0
        1
```

Standard decimal addition

Add 4 and 6 get 10 or 0 and carry 1

Then add 2,9 and 1 to get 12

Standard decimal addition

Add 4 and 6 get 10 or 0 and carry 1

Then add 2,9 and 1 to get 12 or 2 and carry 1

Standard decimal addition

Add 4 and 6 get 10 or 0 and carry 1

Then add 2,9 and 1 to get 12 or 2 and carry 1

Addition

- Add each column together from right
- If bigger than 9, we *carry* over into the next column
- Binary addition is the same, except we carry if the value is greater than one

Binary Addition

```
  1 0 1 1
  0 0 0 1 +
  ---------
```

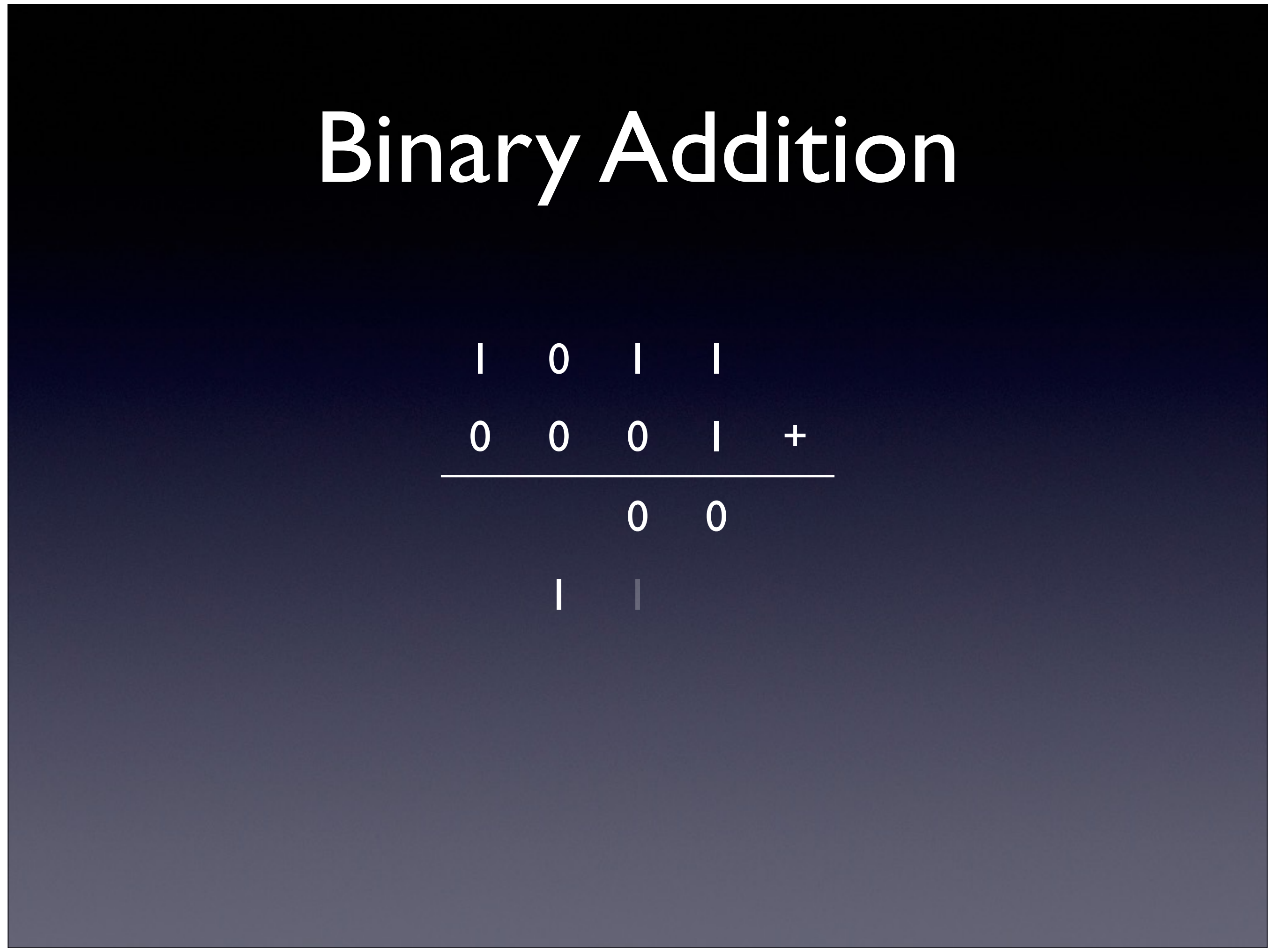
Start on the right add 1 and 1, produces 10 or 0 and carry 1

Binary Addition

1	0	1	1	
0	0	0	1	+
			0	
		1		

Start on the right add 1 and 1, produces 10 or 0 and carry 1

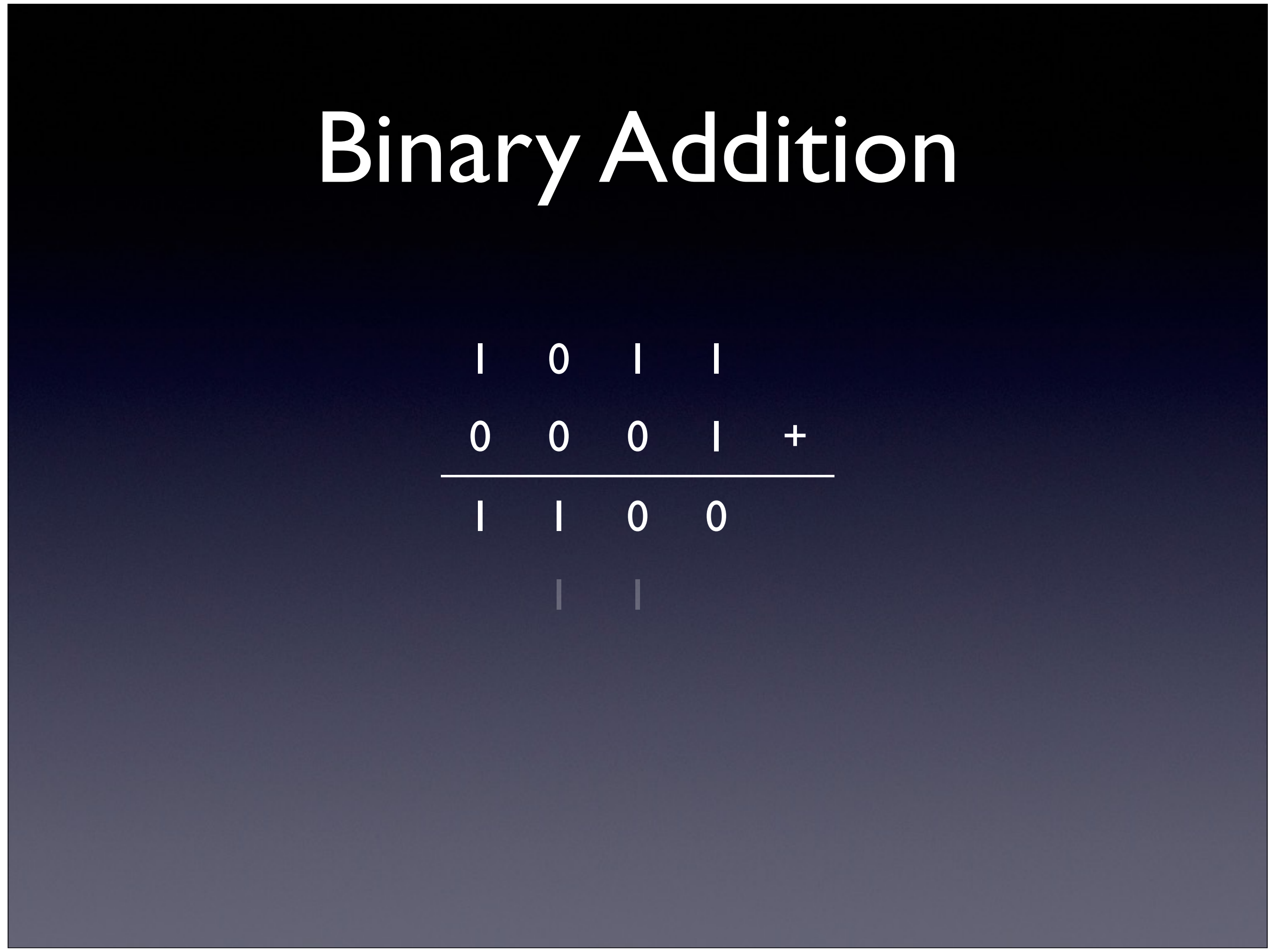
Add 1, 0 and 1 gives 10, or 0 and carry 1



Start on the right add 1 and 1, produces 10 or 0 and carry 1

Add 1, 0 and 1 gives 10, or 0 and carry 1

Add 0 0 and 1 gives 1



Start on the right add 1 and 1, produces 10 or 0 and carry 1

Add 1, 0 and 1 gives 10, or 0 and carry 1

Add 0 0 and 1 gives 1

Adder

- Each column takes in two input bits
- And produces a sum bit and a carry bit
- Can produce a truth table for this…

Adder Truth-Table

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

Adder Truth-Table

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

Adder Truth-Table

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

Adder Truth-Table

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

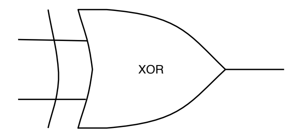
Adder Truth-Table

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

Can now start to think about what logic gates can be used to produce these signals

Carry output is straight-forward… AND gate

eXclusive-OR



A	B	Result
0	0	0
0	1	1
1	0	1
1	1	0

XOR or EOR gate’s truth table looks like this

Symbol used (by engineer’s at least) is ⊕

Demo build in NAND2TETRIS

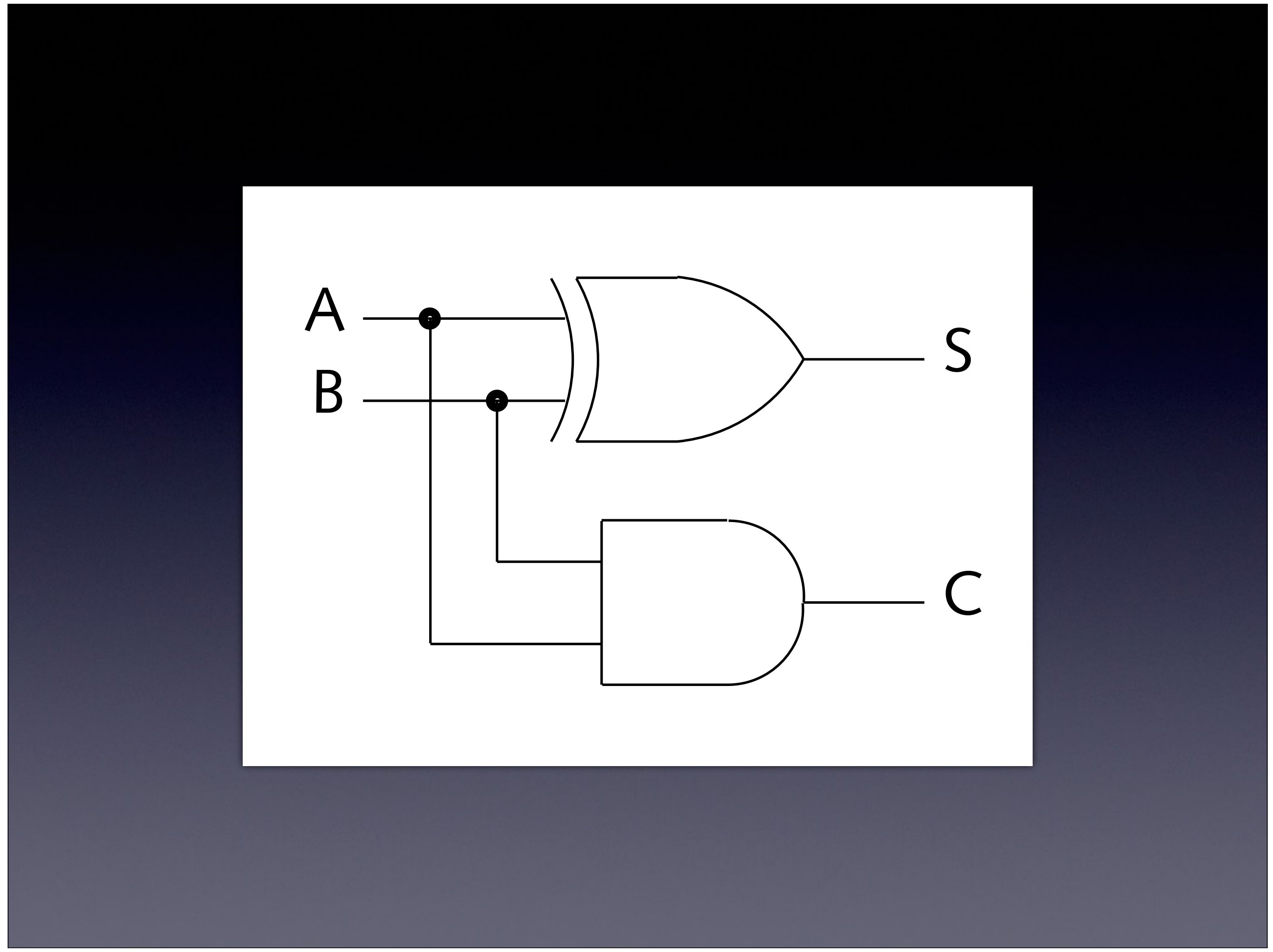
Half-Adder

- Each column takes in two input bits
- And produces a sum bit and a carry bit
- This circuit is known as a half-adder

on the right…

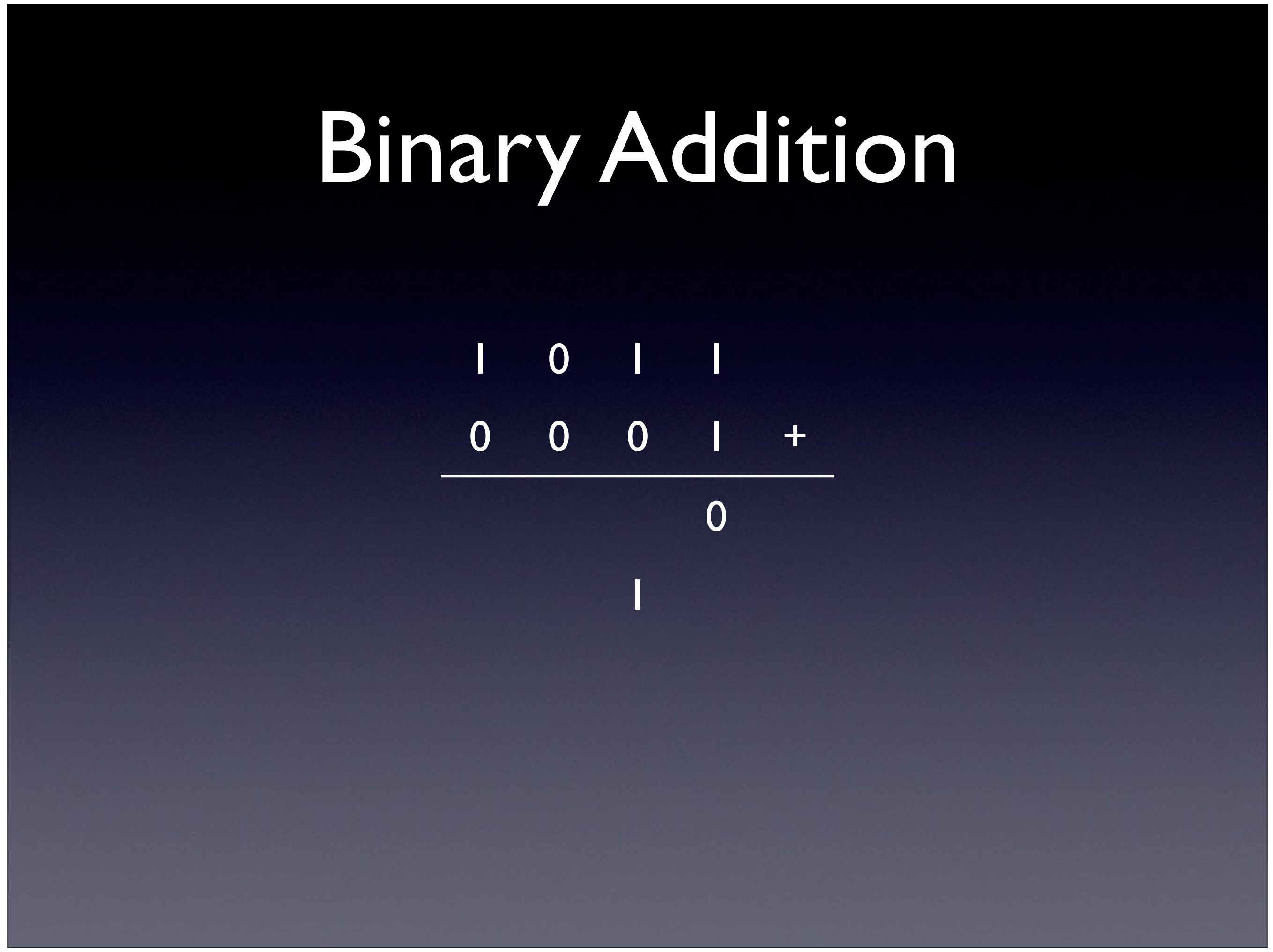
Design Full adder

Equations get more complex


A
B
S
C

Full Adder

- Half-adder only works to add two bits together
- Sometimes we need to add *three* bits
- When we *carry* a bit
- We also need to be able to provide a carry-in bit from the previous column



Start on the right add 1 and 1, produces 10 or 0 and carry 1

Add 1, 0 and 1 gives 10, or 0 and carry 1

C	A	B	S	C
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		
1	1	0		
1	1	1		

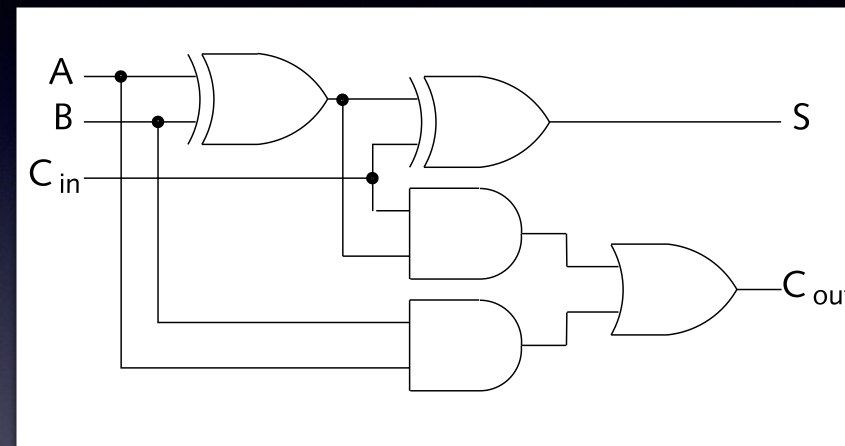
Can now start to think about what logic gates can be used to produce these signals

Carry output is straight-forward… AND gate

C	A	B	S	C
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

Full Adder

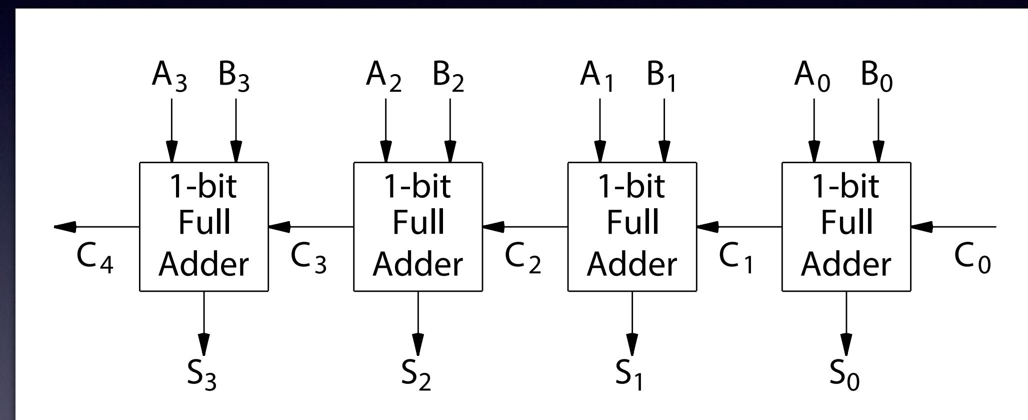
- Circuit described is called a full-adder
- Can combine several of them to add two binary numbers together
- Propagation delay means that it will take some time for the outputs to settle
- So often build adders with several inputs


A
B
C in
S
C out

Chaining Full Adders

- Each adder's C_{out} wired into C_{in} of the next
- n^{th} bit of each input presented to the n^{th} adder's inputs `A` and `B`
- First adder's carry input is `0` (usually)
- Last adder's carry output tells if we've overflowed

Ripple Carry Adder

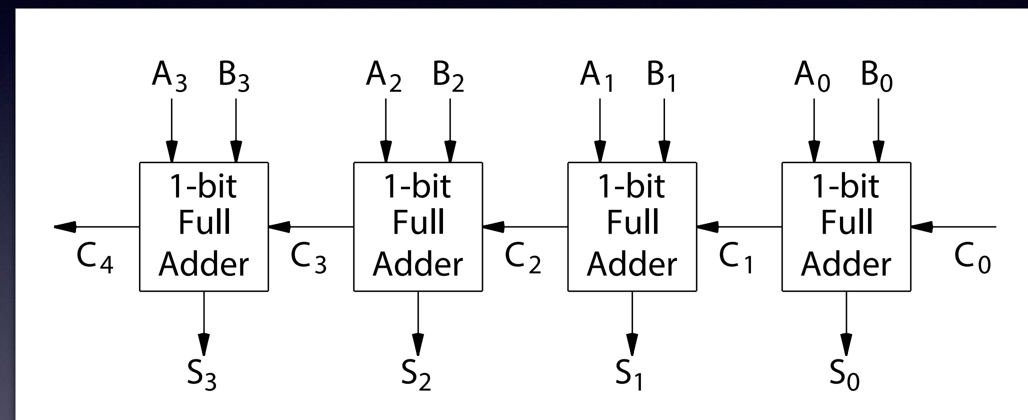


4-bit ripple carry adder

Carry

- Possible for the above Ripple-Carry adder to produce a result bigger than 4-bits
- Hence we still have a carry out
- Adding two *n*-bit numbers can produce an *(n+1)*-bit result
- CPUs preserve the carry bit for you

Ripple Carry Adder



4-bit ripple carry adder