jhucsf.github.io/fall2019

Addition and Subtraction

Philipp Koehn

4 September 2019

addition

1-Bit Adder

- Let's start simple: Adding two 1-Bit numbers

1-Bit Adder

- Let's start simple: Adding two 1-Bit numbers

- Truth table

A	B	A+B
0	0	0

1-Bit Adder

- Let's start simple: Adding two 1-Bit numbers

- Truth table

A	B	A+B
0	0	0
0	1	1

1-Bit Adder

- Let's start simple: Adding two 1-Bit numbers

- Truth table

A	B	A+B
0	0	0
0	1	1
1	0	1

1-Bit Adder

- Let's start simple: Adding two 1-Bit numbers

- Truth table

A	B	A+B
0	0	00
0	1	01
1	0	01
1	1	10

Really 2 Operations

- Truth table for "position 0" bit

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	0

Really 2 Operations

- Truth table for "position 0" bit

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	0

xor

Really 2 Operations

- Truth table for "position 0" bit

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	0

xor

- Truth table for carry bit

A	B	A+B carry
0	0	0
0	1	0
1	0	0
1	1	1

Really 2 Operations

- Truth table for "position 0" bit

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	0

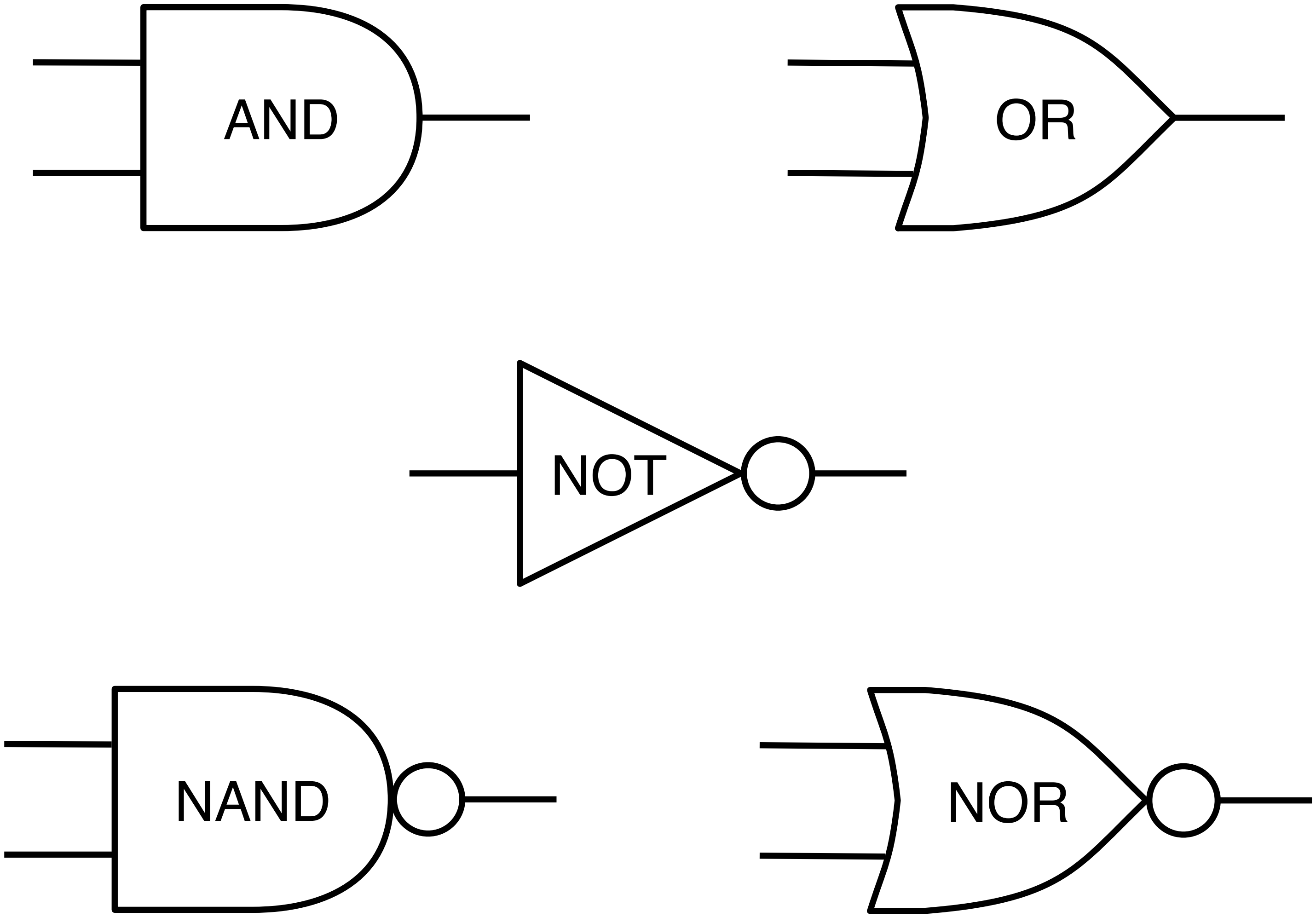
xor

- Truth table for carry bit

A	B	A+B carry
0	0	0
0	1	0
1	0	0
1	1	1

and

Reminder: Basic Gates



Circuits

- "Position 0" bit

A	B	OUT0
0	0	0
0	1	1
1	0	1
1	1	0

Circuits

- "Position 0" bit

A	B	OUT0
0	0	0
0	1	1
1	0	1
1	1	0

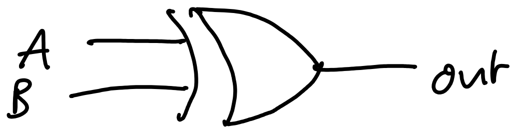
xor

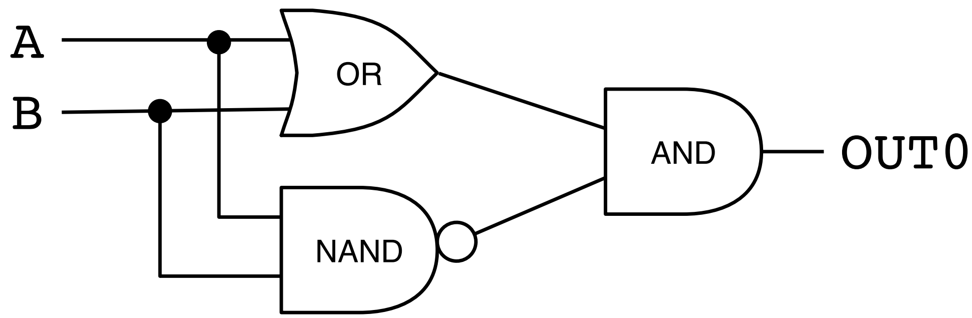
Circuits

- "Position 0" bit

A	B	OUT0
0	0	0
0	1	1
1	0	1
1	1	0

xor





- Carry bit

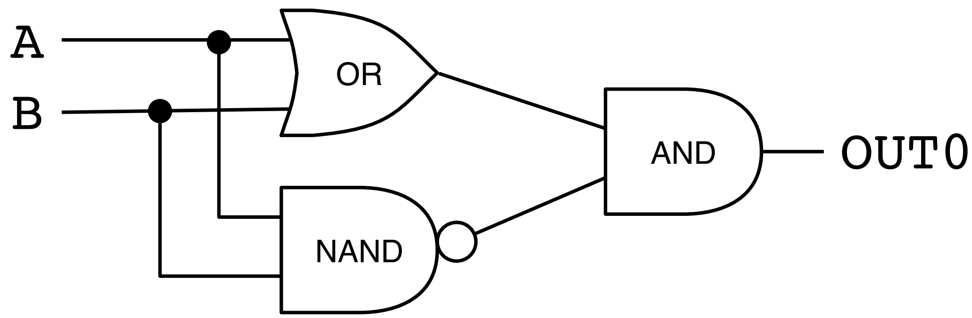
A	B	OUTC
0	0	0
0	1	0
1	0	0
1	1	1

Circuits

- "Position 0" bit

A	B	OUT0
0	0	0
0	1	1
1	0	1
1	1	0

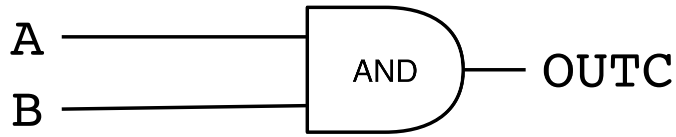
xor



- Carry bit

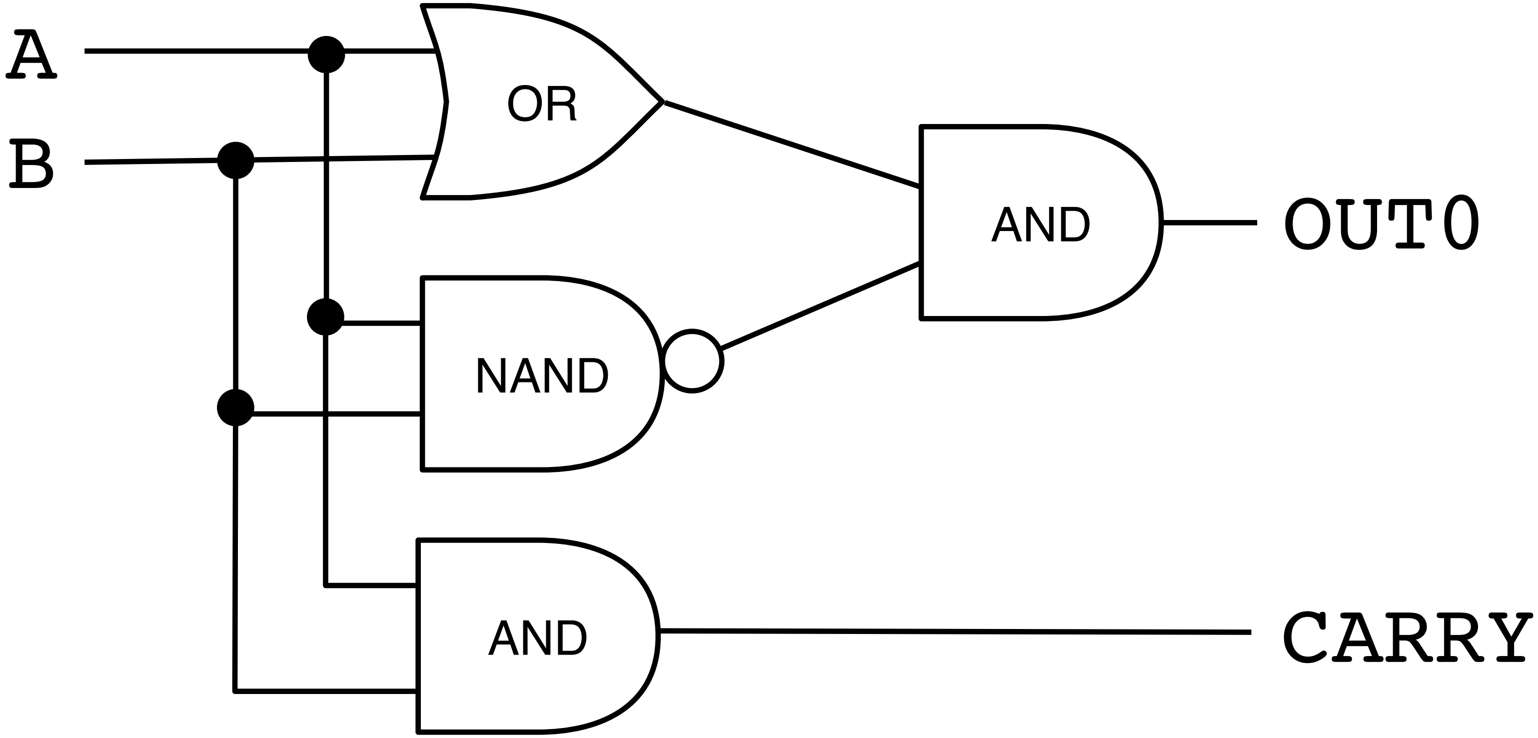
A	B	OUTC
0	0	0
0	1	0
1	0	0
1	1	1

and



Putting them Together

Addition!



N-Bit Addition


```
 11
+11
---

---
```

N-Bit Addition


```
 11
+11
---
 1
---
  0
```

1+1 = 0, carry the 1

N-Bit Addition


```
 11
+11
---
11
---
 10
```

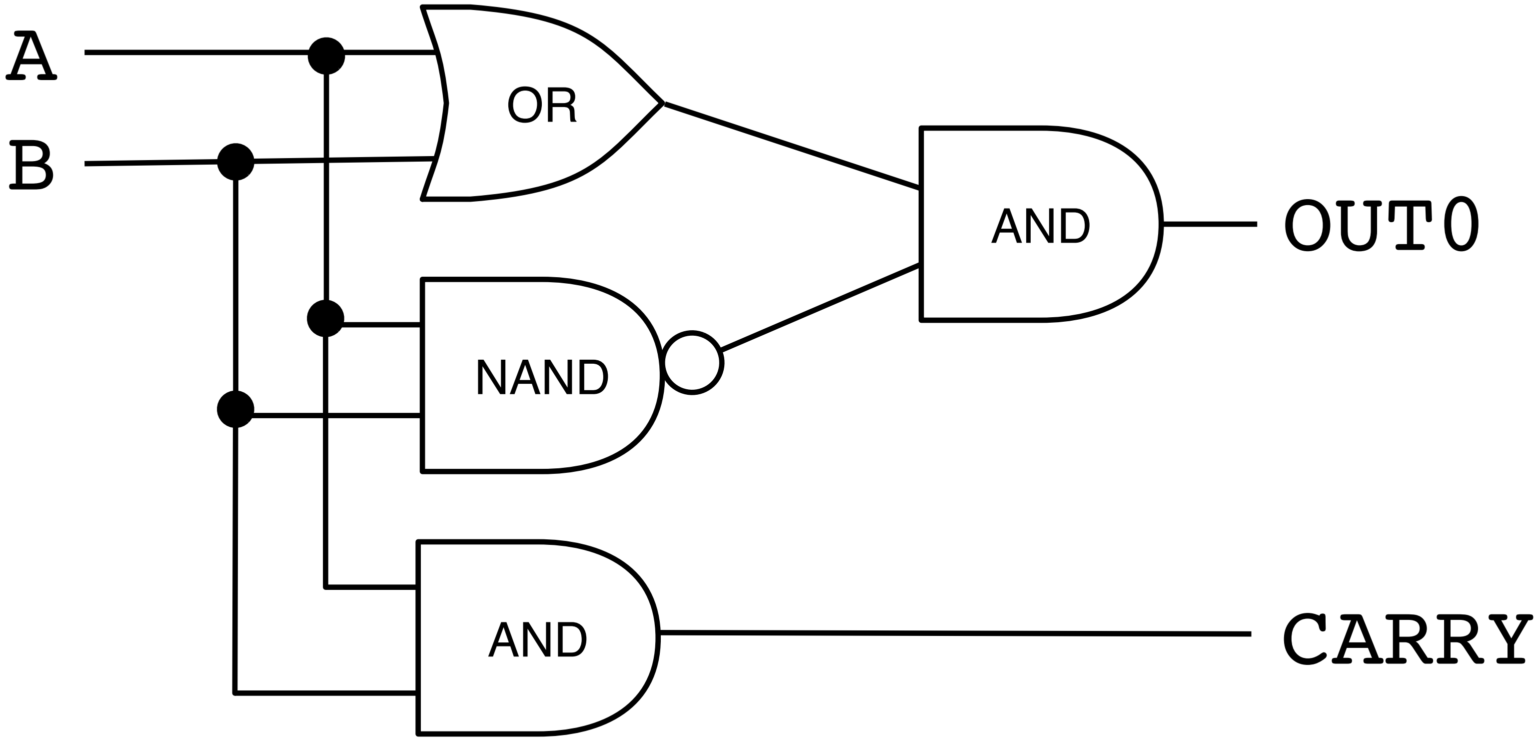
```
1+1+1 = 1, carry the 1
```

N-Bit Addition


```
 11
+11
---
11
---
110
```

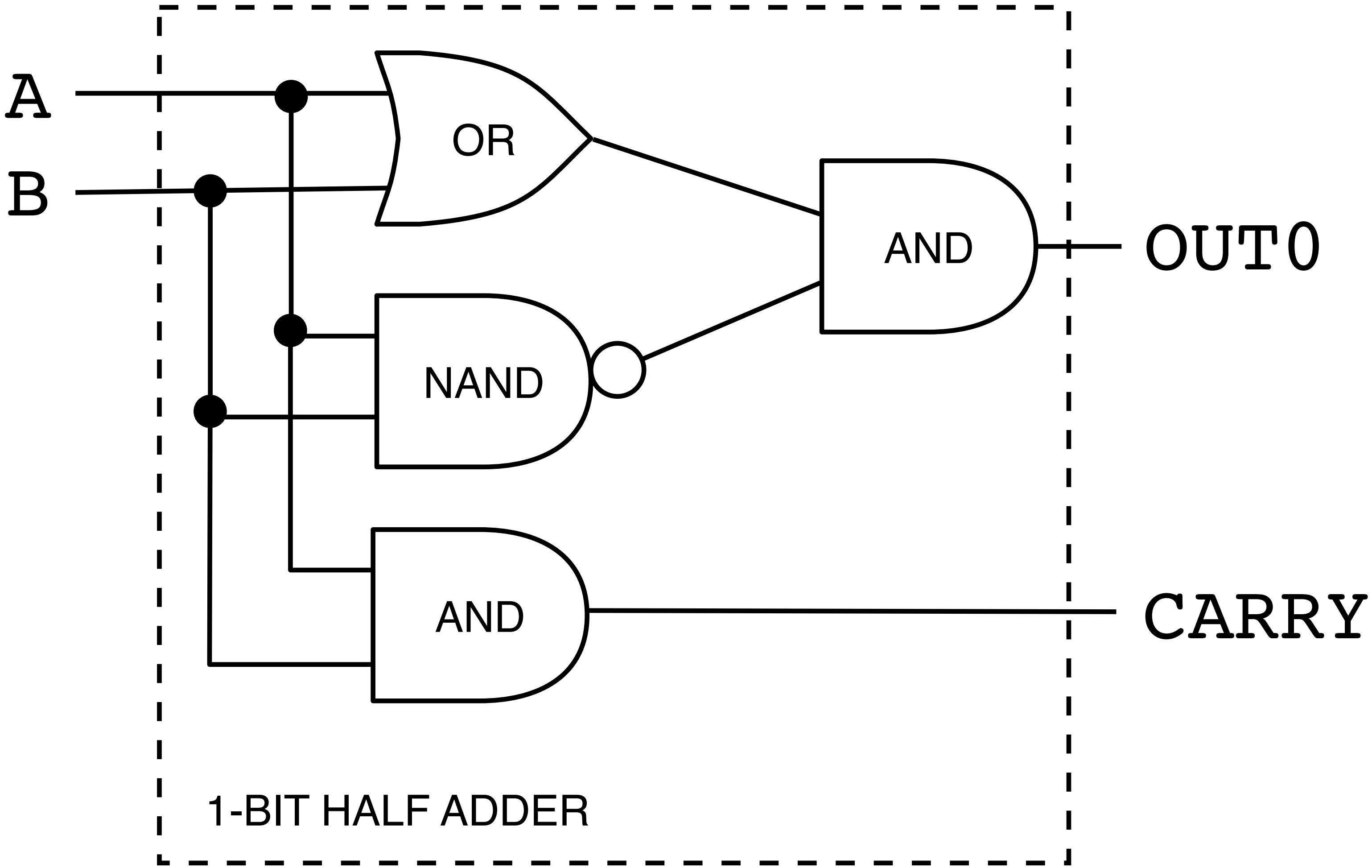
copy carry bit

1-Bit Adder

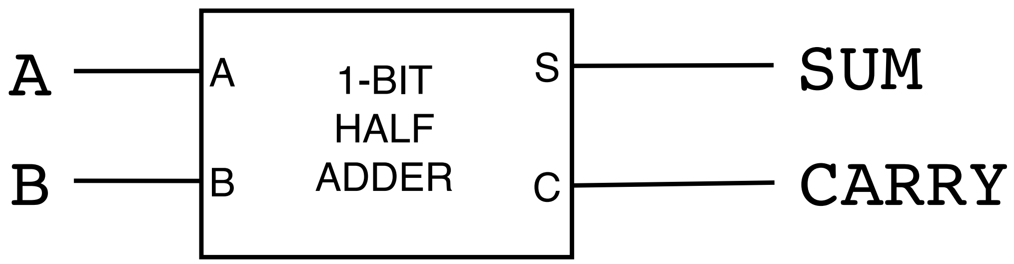


Our adder cannot handle carry as input yet

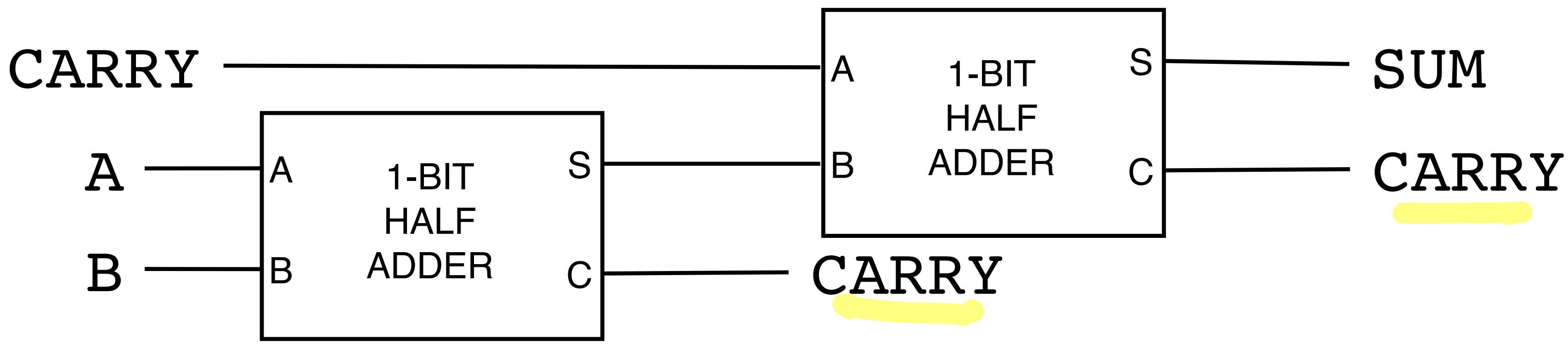
Half 1-Bit Adder



Building a 1-Bit Full Adder

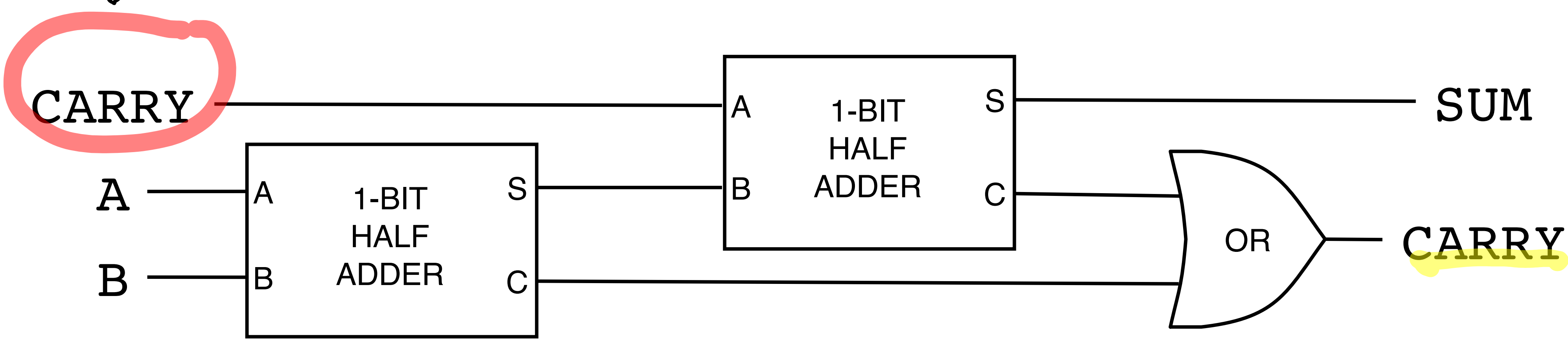


Building a 1-Bit Full Adder

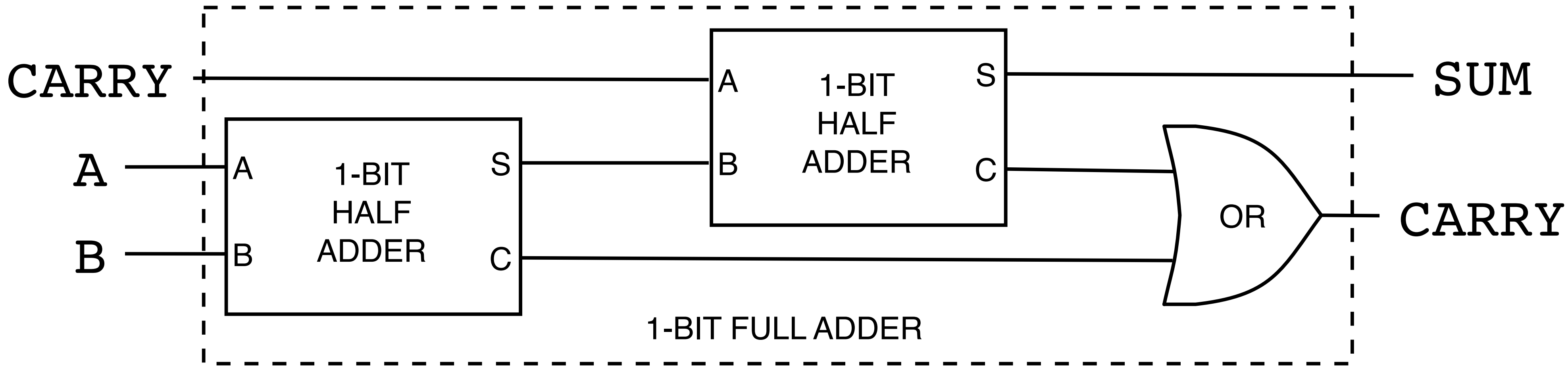


Building a 1-Bit Full Adder

comes from previous column



1-Bit Full Adder



N-Bit Full Adder

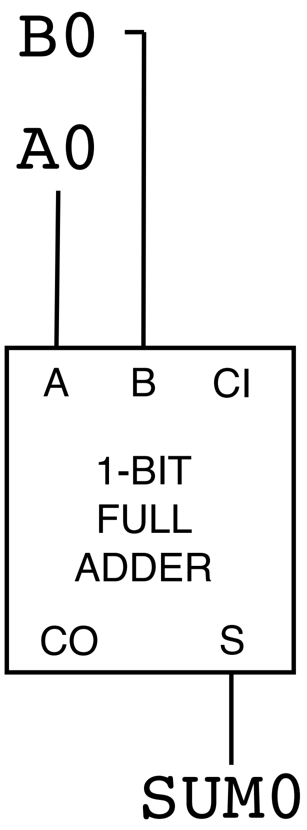

```
 11
+11
---

---
```

N-Bit Full Adder


```
 11
+11
---
 1
---
  0
```

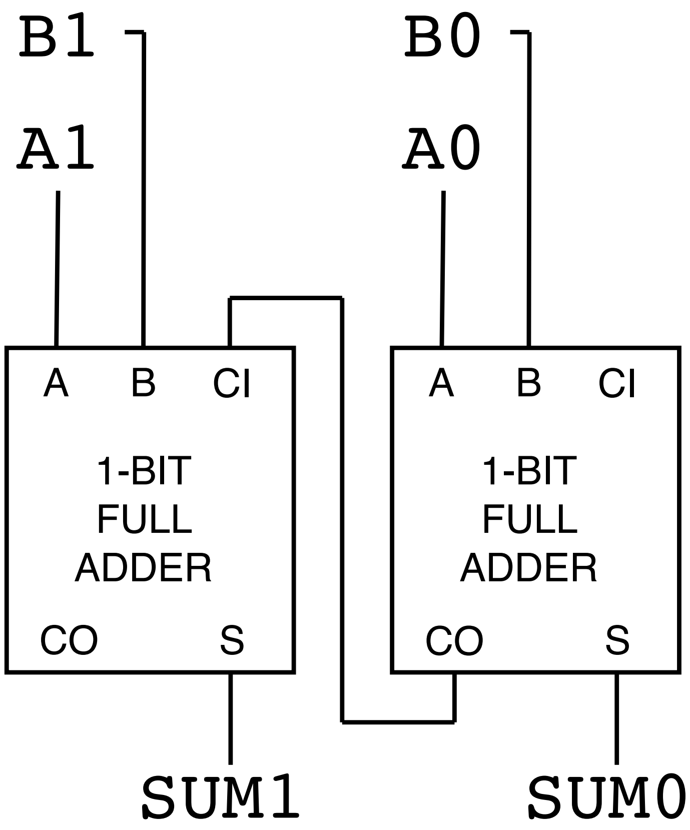
N-Bit Full Adder


```
 11
+11
---
 1
---
  0
```



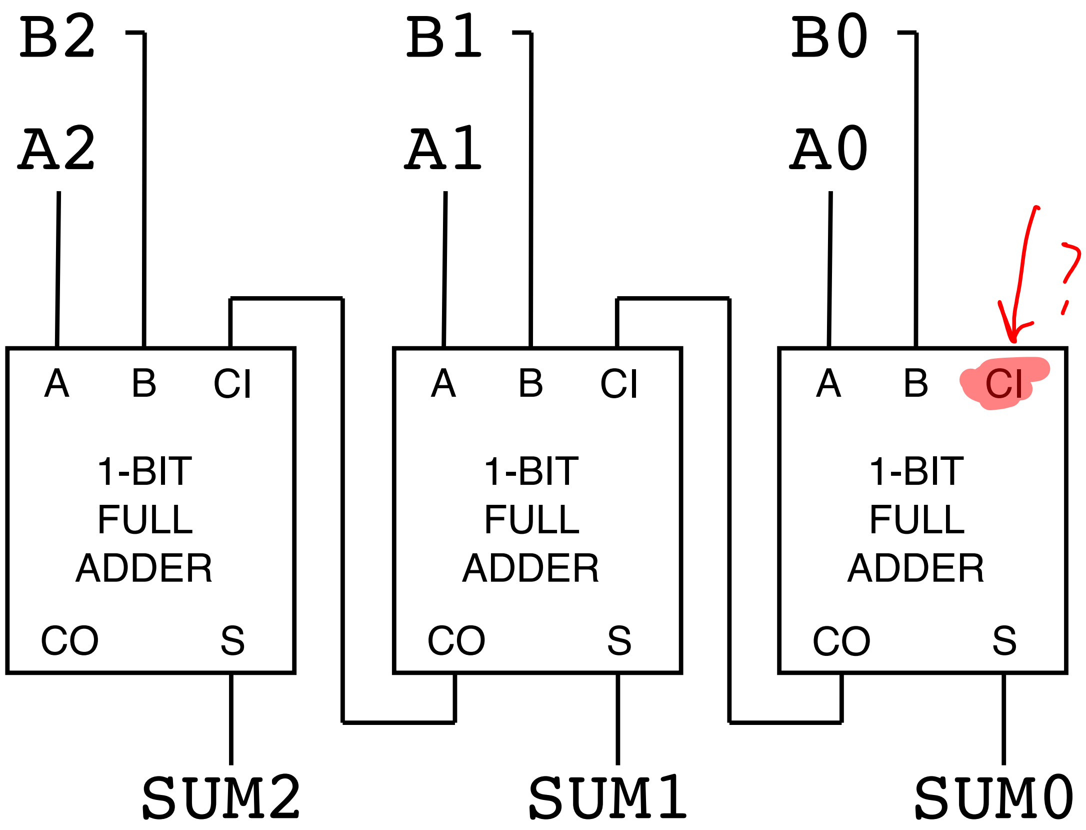
N-Bit Full Adder


```
 11
+11
---
11
---
 10
```



N-Bit Full Adder

and
so
on
...



subtraction

First, a Trick

"minuend" "subtrahend"

A − B = C

"difference"

- Normally, we subtract like this:

```
 253
-176
----
 11
----
  77
```

Computing the Inverse

- Now we use the inverse of the subtrahend

```
 999
-176
----
 823
```

"9's complement"

Subtraction by Addition

- This allows us to carry our subtraction by addition

```
  253
+ 823
-----
 1076
```

Subtraction by Addition

- This allows us to carry our subtraction by addition

```
  253
+ 823
-----
 1076
```

- Well, with minor corrections

```
 1076
+   1
-1000
-----
   77
```

Also Works in Binary


```
Original problem                 253          11111101
                               - 176        - 10110000
                               -----        ----------
                                  77          01001101
```

Also Works in Binary


```
Original problem          253      11111101
                        - 176    - 10110000
                        -----    ----------
                           77      01001101

Inverse of subtrahend     823      01001111
```

invert bits of subtrahend

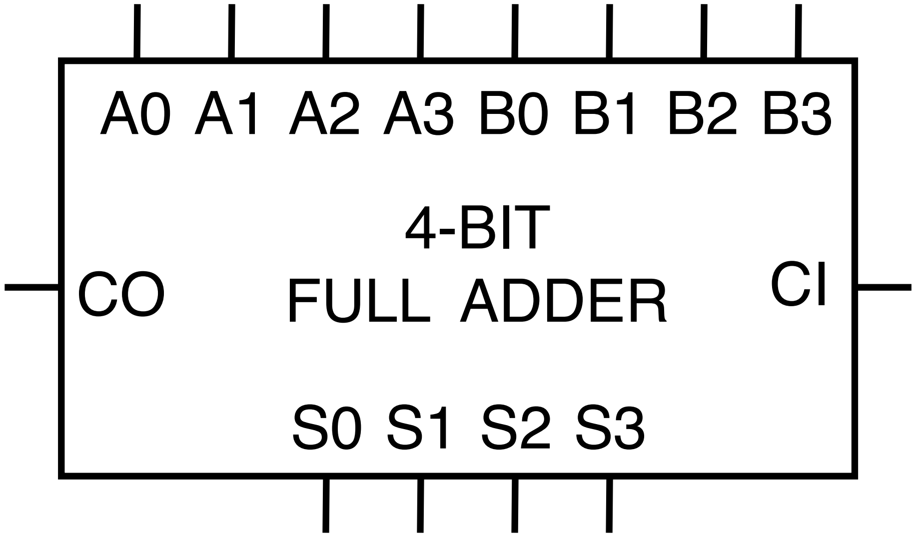
Also Works in Binary


```
Original problem             253       11111101
                           - 176     - 10110000
                           -----     ----------
                              77       01001101

Inverse of subtrahend        823       01001111

Addition                     253       11111101
                           + 823     + 01001111
                           -----     ----------
                            1076      101001100
```

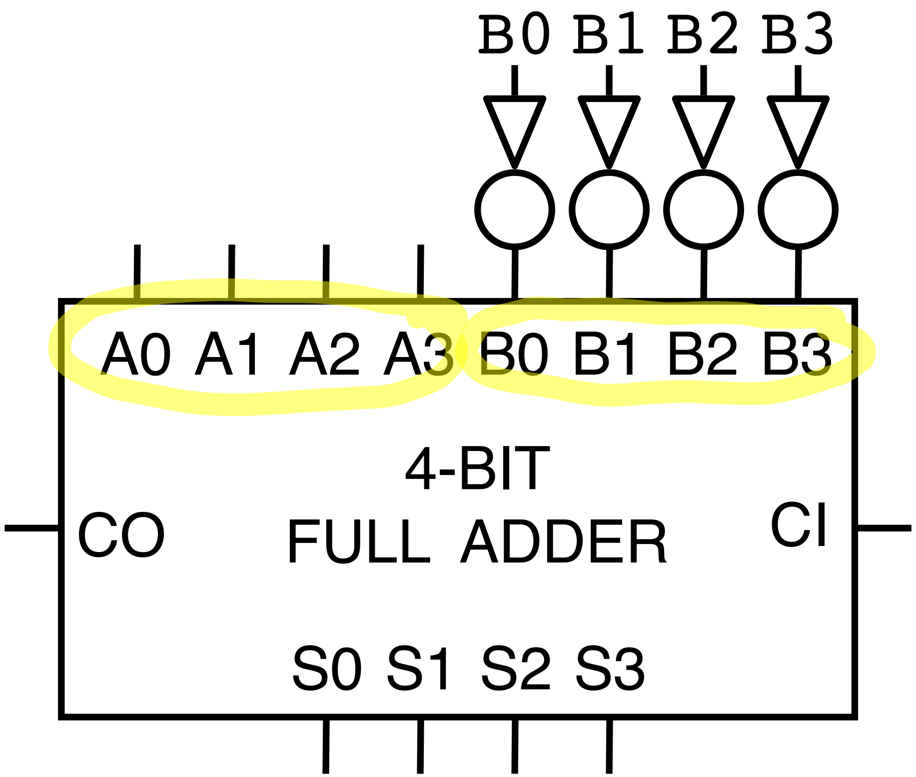
Also Works in Binary


```
Original problem              253        11111101
                            - 176      - 10110000
                            -----      ----------
                               77        01001101

Inverse of subtrahend         823        01001111

Addition                      253        11111101
                            + 823      + 01001111
                            -----      ----------
                             1076       101001100

Corrections                 +   1      +        1
                            -1000      -100000000
                          ----------   ----------
                             77          01001101
```

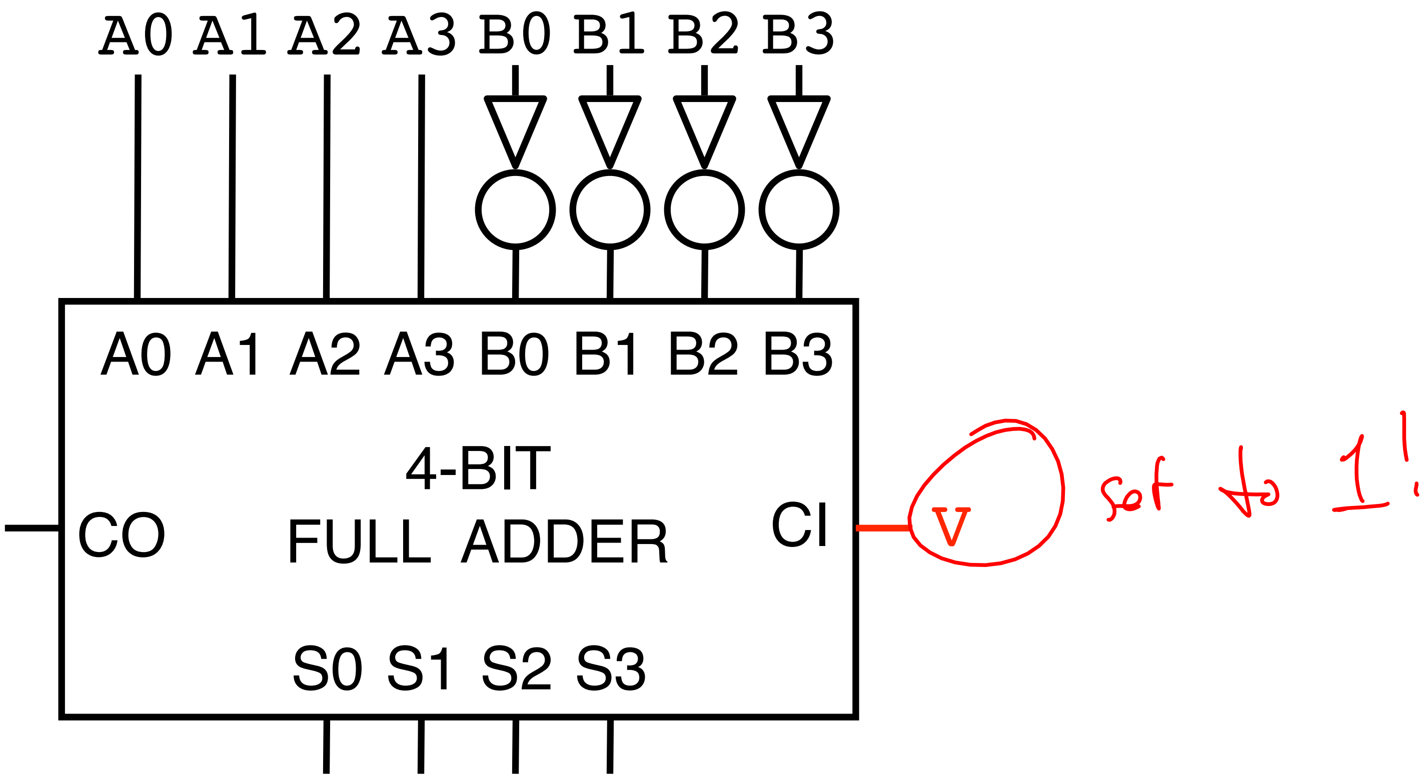
Start with N-Bit Adder



Invert Bits of Subtrahend

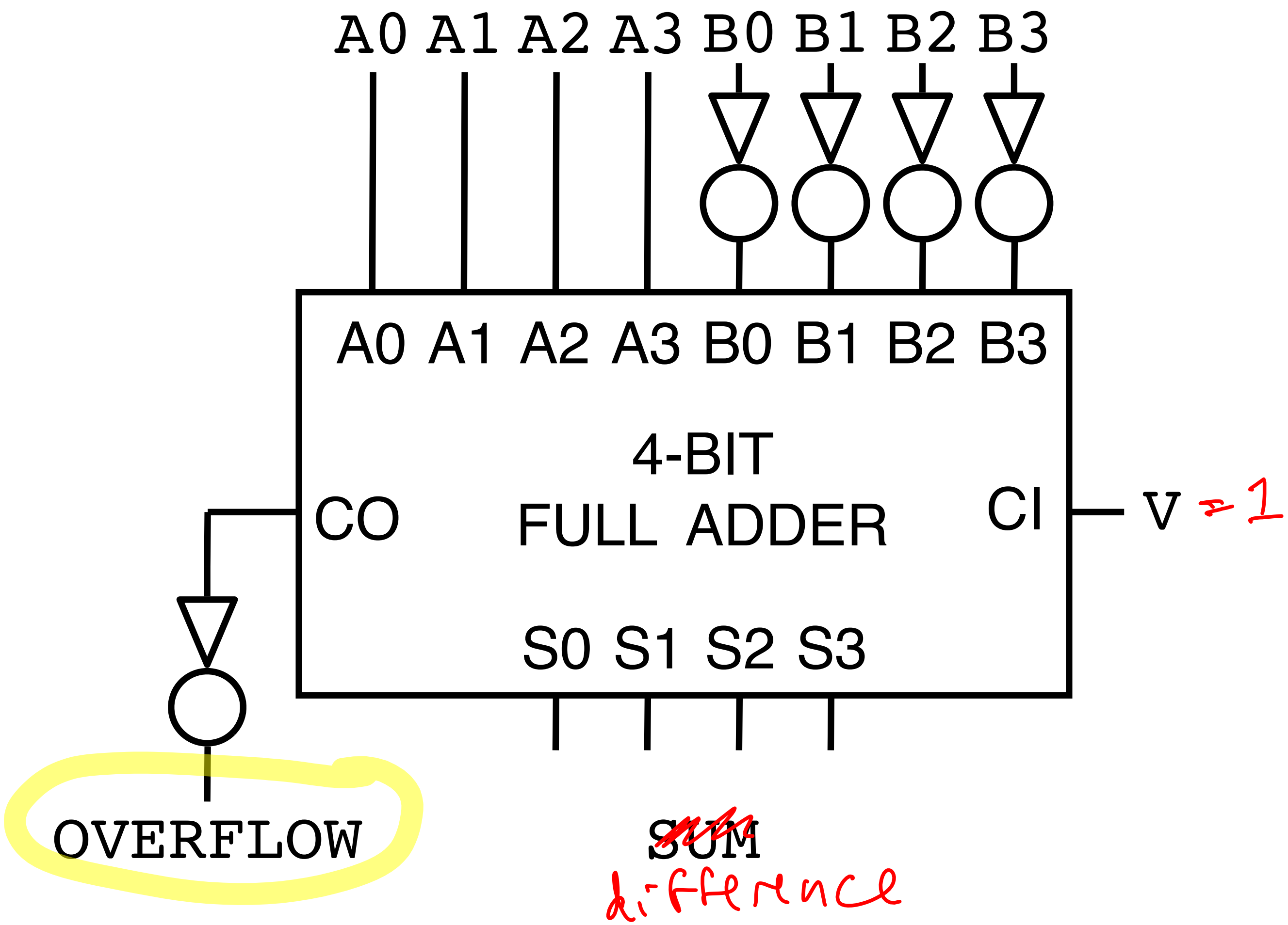


Add One



Trick: add one as carry in

Invert Overflow --- DONE



unifying

addition and subtraction

machines

Goal

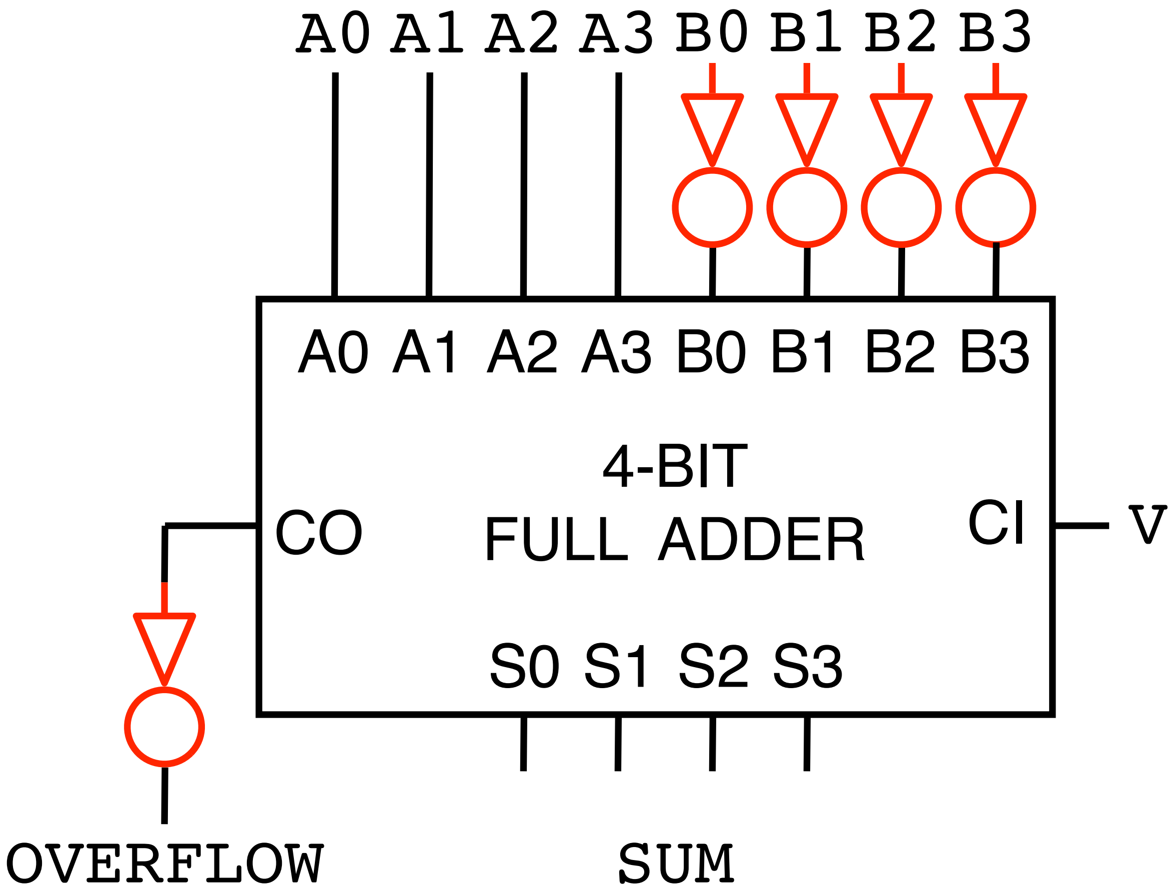
- Not two machines for addition and subtraction

⇒ Combined adder and subtractor

- Input: A, B, and subtraction flag SUB

- Output
  - if SUB=0: A+B
  - if SUB=1: A-B

NOT only if SUB



NOT only if SUB

- Truth table

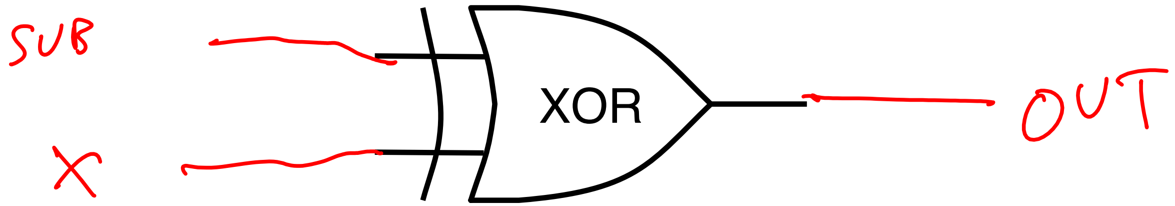
SUB	X	OUT
0	0	0
0	1	1
1	0	1
1	1	0

NOT only if SUB

- Truth table

SUB	X	OUT
0	0	0
0	1	1
1	0	1
1	1	0

- Looks like XOR



Combined Machine