January 28, 2026

Lesson Plan

- Gates
- Binary Addition
- Half adder
- Full adder
- Ripple adder

RECAP

- Number representations.

Gates

* Inverter: NOT gate

A	$\neg A$ / A'
0	1
1	0

* OR gate

A	B	A ∪ B
0	0	0
0	1	1
1	0	1
1	1	1

* NAND gate

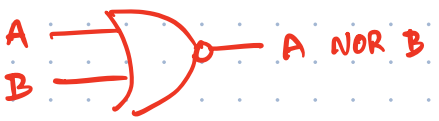
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

* AND gate

A	B	A ∩ B
0	0	0
0	1	0
1	0	0
1	1	1

* NOR gate

A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0



* XOR gate

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

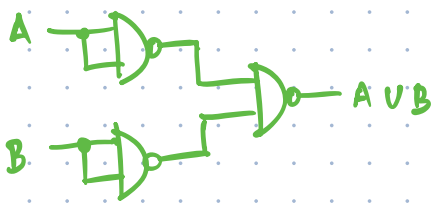
These three can express any Boolean expression.

- NAND and NOR are universal gates.

(i) 

(ii) 

(iii) 

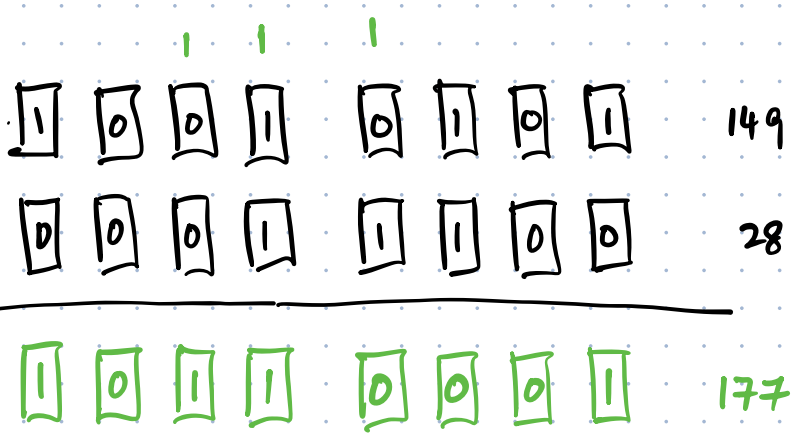


(iv) TO DO: 

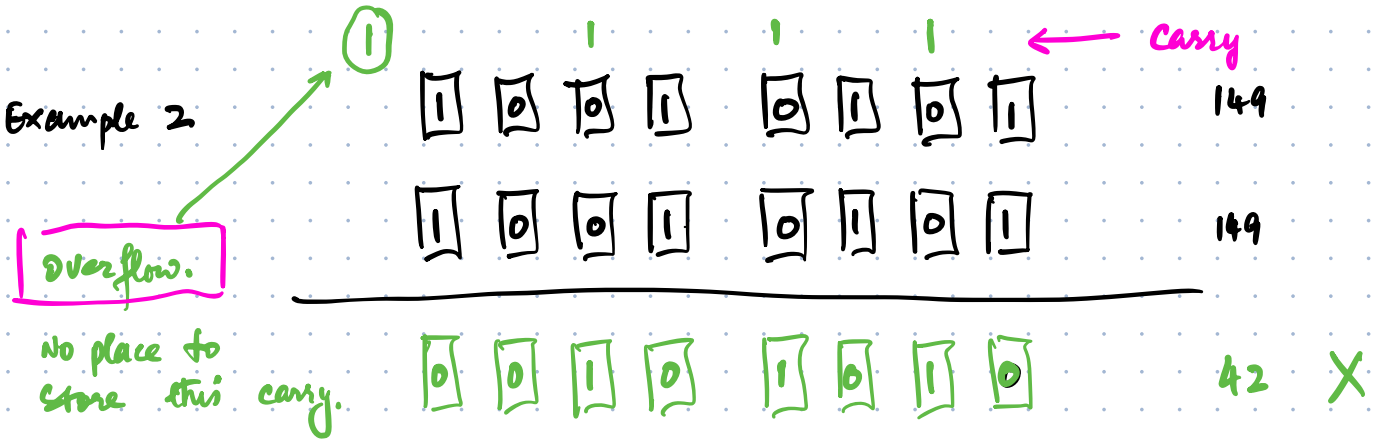
★ Binary Arithmetic

Carry →

Example 1



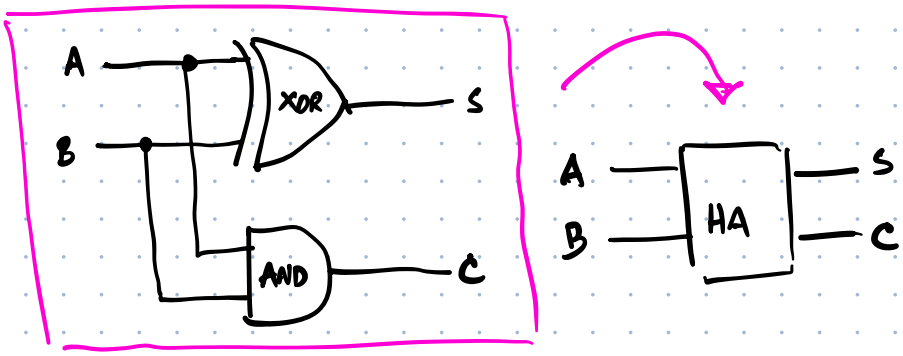
Example 2



Adder Circuits

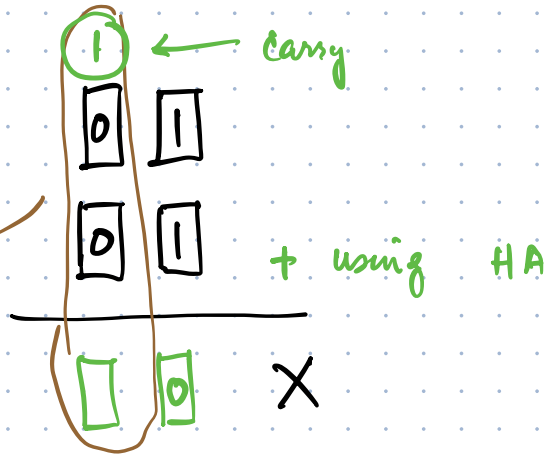
(i) Half adder : adds two bits

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



Thought Experiment

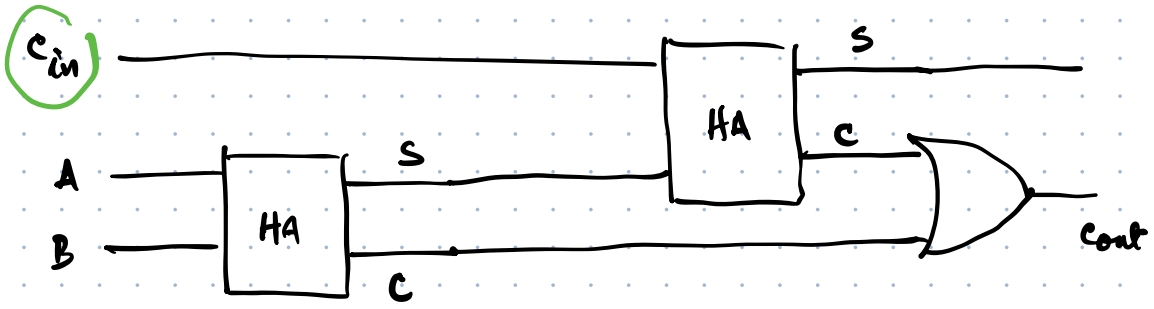
HA cannot deal with three bits

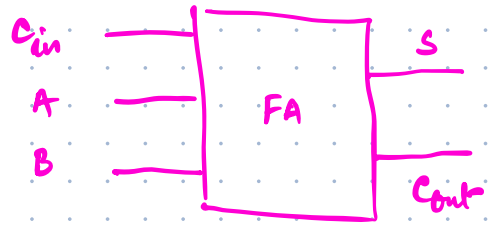


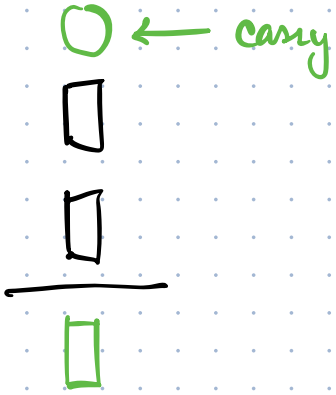
(ii)

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

C_{in}	S	S'	C'
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



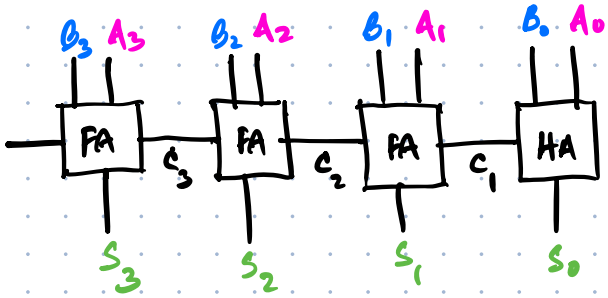




C_{in}	A	B	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Q. How do we add 4-bit numbers?

↓ Ripple Adder



If C_{out} is not 0, this indicates an overflow.

"SLOW" this would need 4 clock cycles to add two 4-bit numbers. What if we have 64-bit numbers ?!

Alternate: fast adders.

* Subtraction

$$A - B = A + (-B)$$

↑ we know how to represent negative numbers.

Ex. 1. Compute 1-2 using 4-bit numbers

Sign bit

```
 0 0 0 1    (+1)
 1 0 1 0    (-2)
-----------
 1 0 1 1    (-3)   X
```

Ex. 2. Compute 1-7 using 4-bit numbers

* Represent numbers using excess-7 coding.

```
 1 0 0 0    (+1)
 0 0 0 0    (-7)
-----------
 1 0 0 0    (+1)   X
```

→ A new way of encoding -ve numbers:

$$A + (-B) = A - B$$