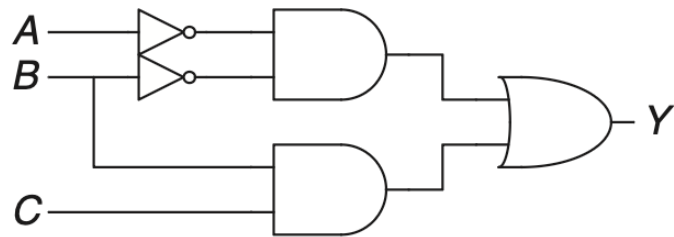


ENR-325/325L Principles of Digital Electronics and Laboratory

Xiang Li
Fall 2025

K-map is for Boolean logics, not engineering

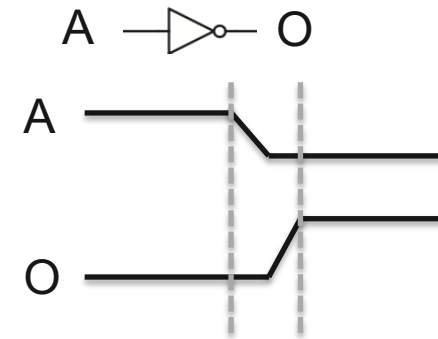


		AB			
		00	01	11	10
C	0	1	0	0	0
	1	1	1	1	0

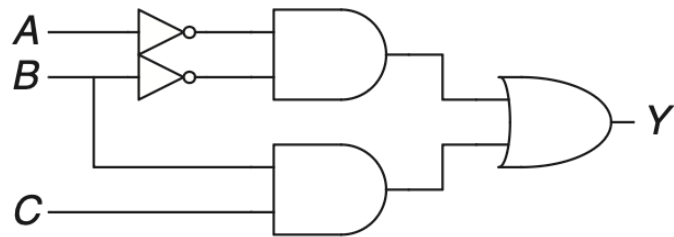
$$Y = \overline{A}\overline{B} + BC$$

Figure 2.75 Circuit with a glitch

Why glitch? Because in real-life every flip takes time:



K-map is for Boolean logics, not engineering



Y C	AB			
	00	01	11	10
0	1	0	0	0
1	1	1	1	0

$$Y = \overline{A}\overline{B} + BC$$

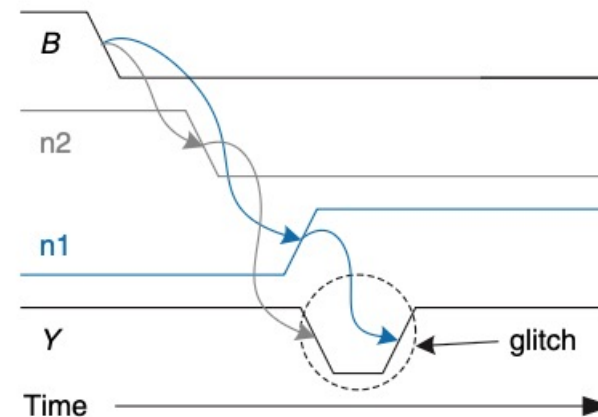
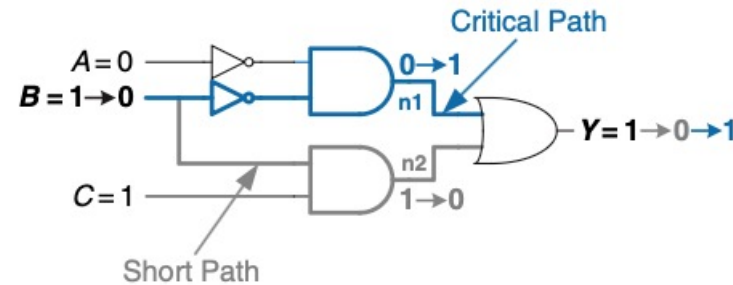


Figure 2.75 Circuit with a glitch

Hamming codes can be done in the EE way

- Before that, we need to acquire some basic skillsets.

Pre-step: Data forms

Step 1: Data manipulation

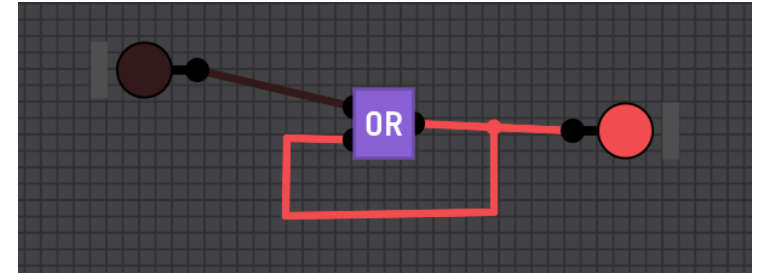
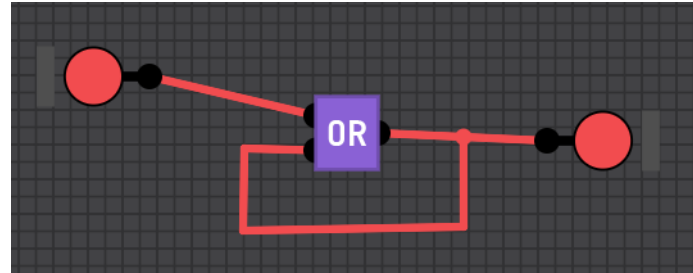
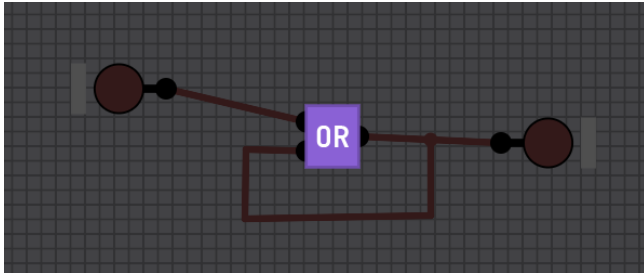
Step 2: Information storage

Step 3: Interface

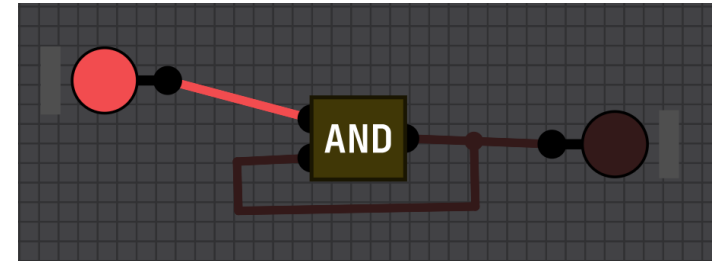
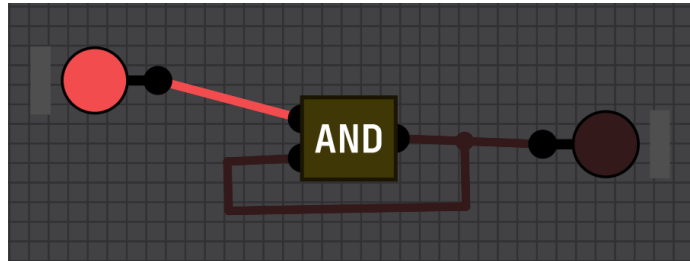
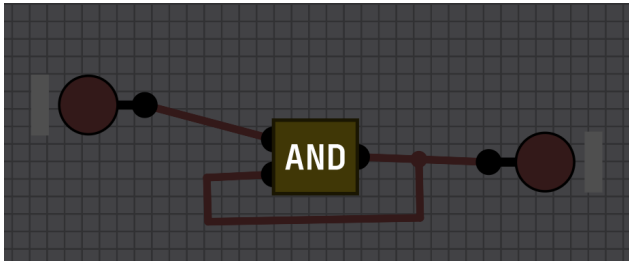
Step 1: data manipulation, continued

- Self-looped gates -> latches

Once on, always on!



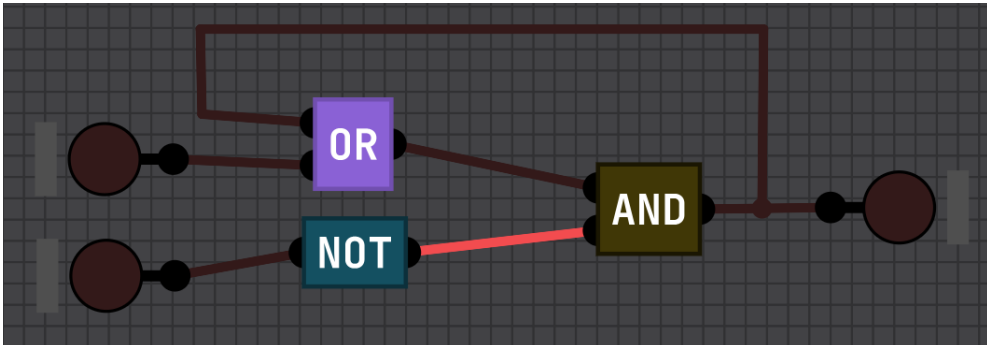
Always off, meh.



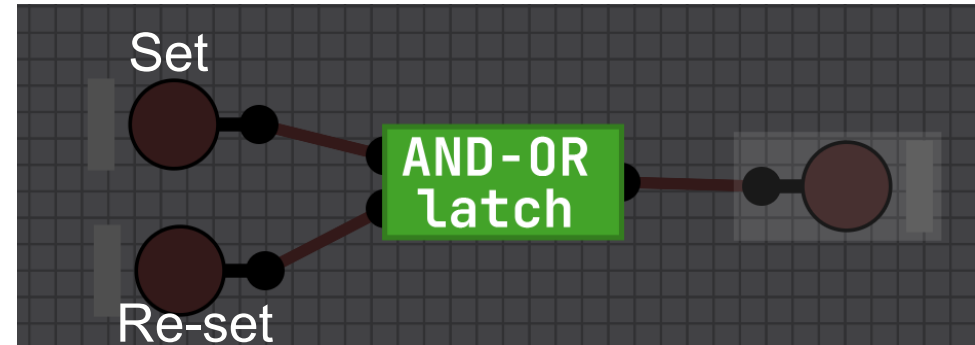
COE COLLEGE®

Step 1: data manipulation, continued

- Make the AND-OR latch a bit useful: it stores(writes/clears) 1 bit, kind a.

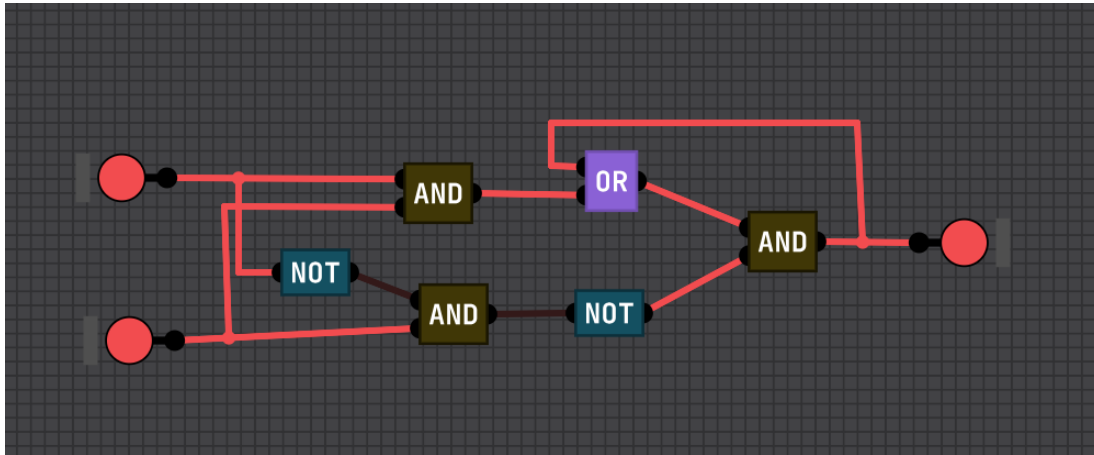


An abstraction of AND-OR latch

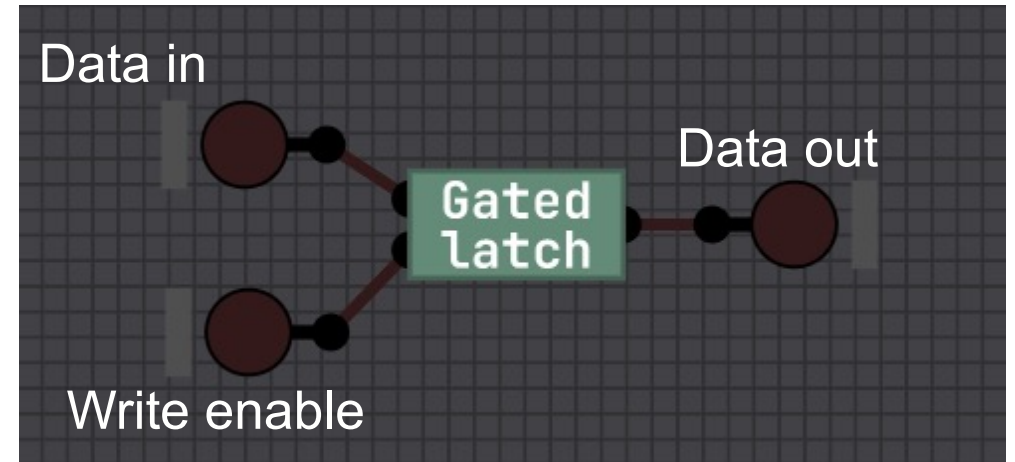


Step 2: Detour to information storage

This is a Gated latch, it stores(writes/clears) 1 bit slightly better.

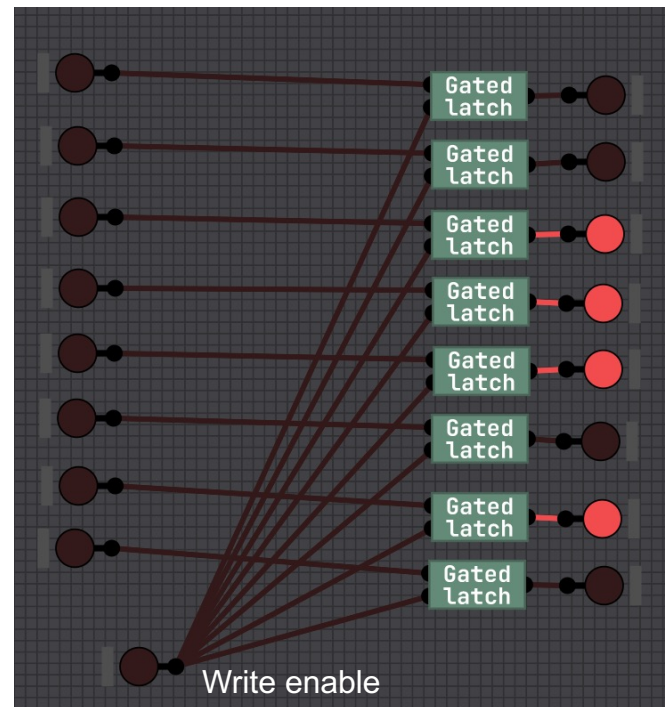


An abstraction of gated latch.

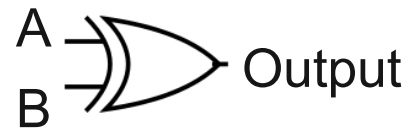
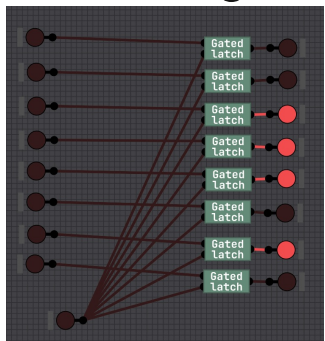
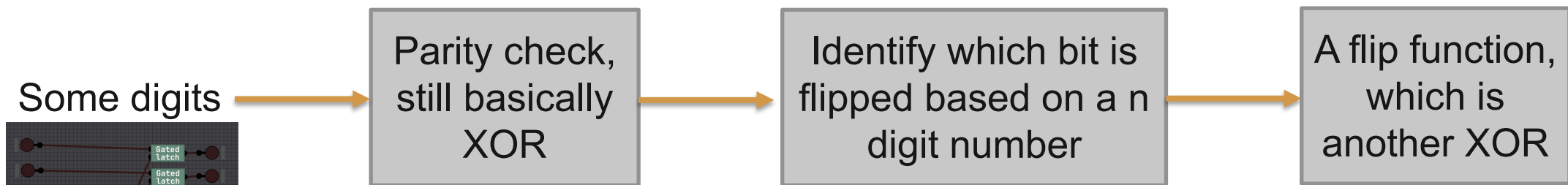


Step 2: Information storage, all done!

Scaling up to 8, we got an 8 bit storage (register).



So how about Hamming **decoder** (15,11)?



XOR gate truth table

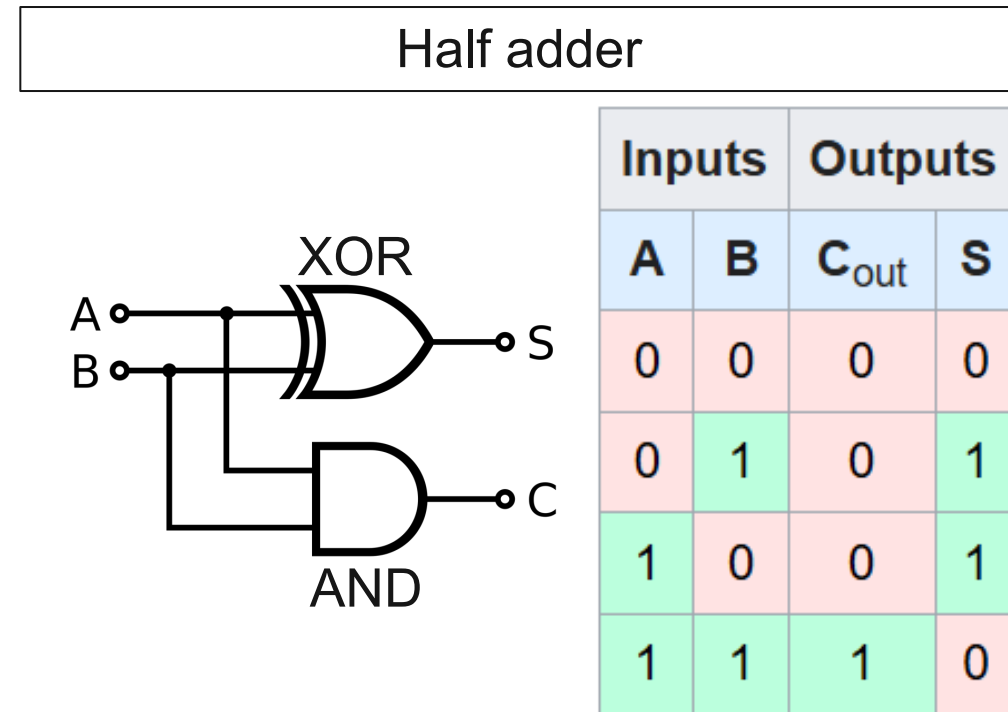
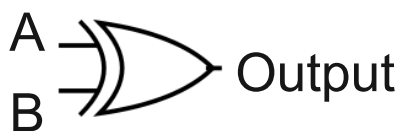
Input		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

XOR gate can do so much more: addition

- XOR gate is almost what we need:

XOR gate truth table

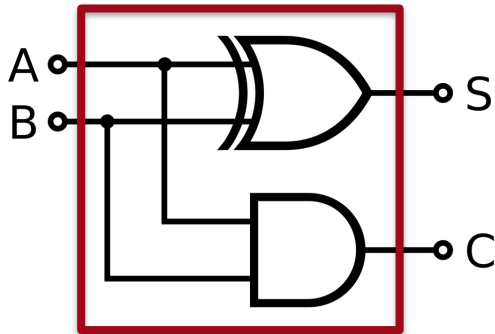
Input		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0



Step 1: back to data manipulation

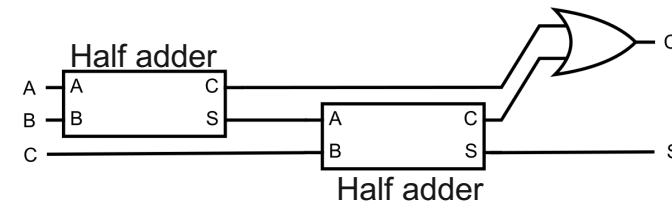
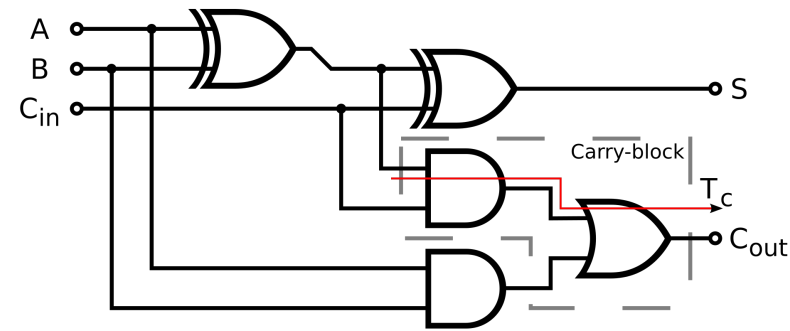
- Boolean operations: 3rd gen:

Half adder



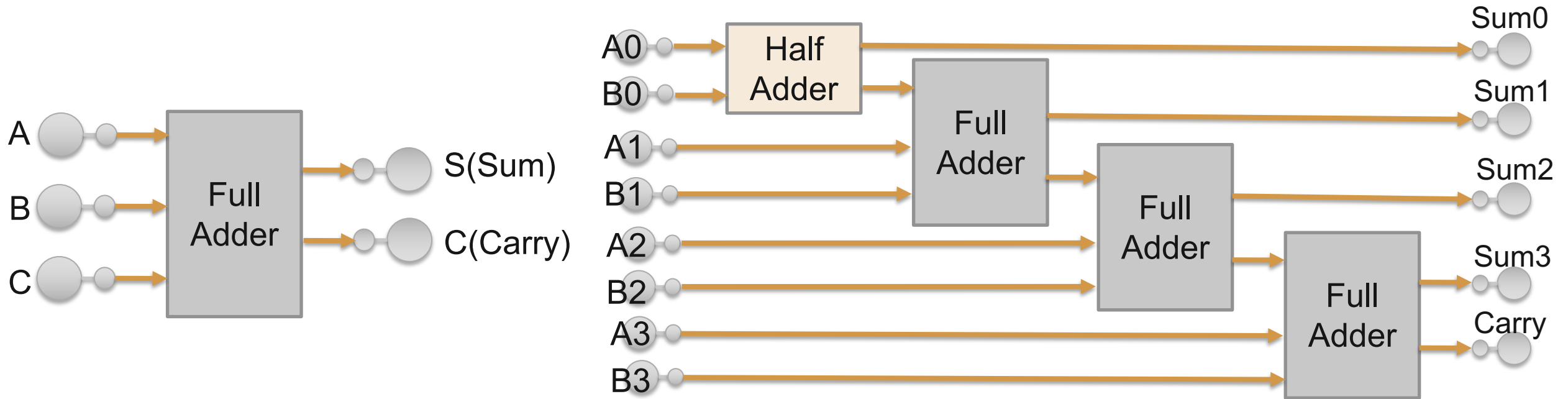
Inputs		Outputs	
A	B	C _{out}	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Full adder



Inputs			Outputs	
A	B	C _{in}	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

More abstraction: full adder circuits



Let's realize subtraction:

Step 1: create "negative number"

Step 2: add negative number

1's compliment

-5	1010
-4	1011
-3	1100
-2	1101
-1	1110
-0	1111
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101

Just invert every 1 and 0, and it almost works.

1st bit no longer carry value, but signs

Let's realize subtraction:

Step 1: create "negative number"

Step 2: add negative number

	1's compliment		2's compliment
-5	1010	+1	1011
-4	1011	+1	1100
-3	1100	+1	1101
-2	1101	+1	1110
-1	1110	+1	1111
-0	1111	+1	10000
0	0000		0000
1	0001		0001
2	0010		0010
3	0011		0011
4	0100		0100
5	0101		0101

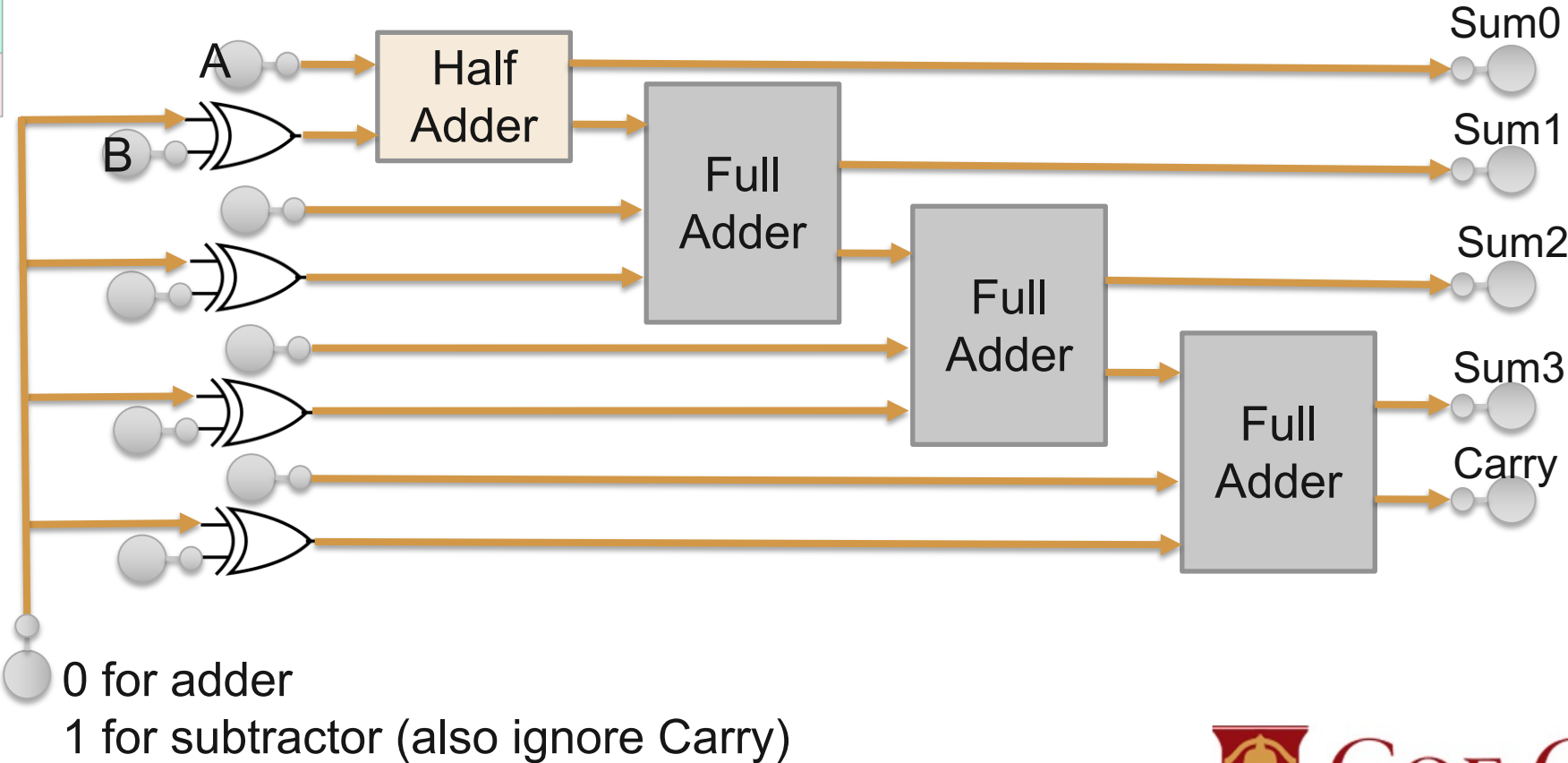
Sign bits

- It works.
- Instead of subtraction, we do addition.

XOR gate truth table

Input		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

4 bit adder and subtractor



Step 3: Interface

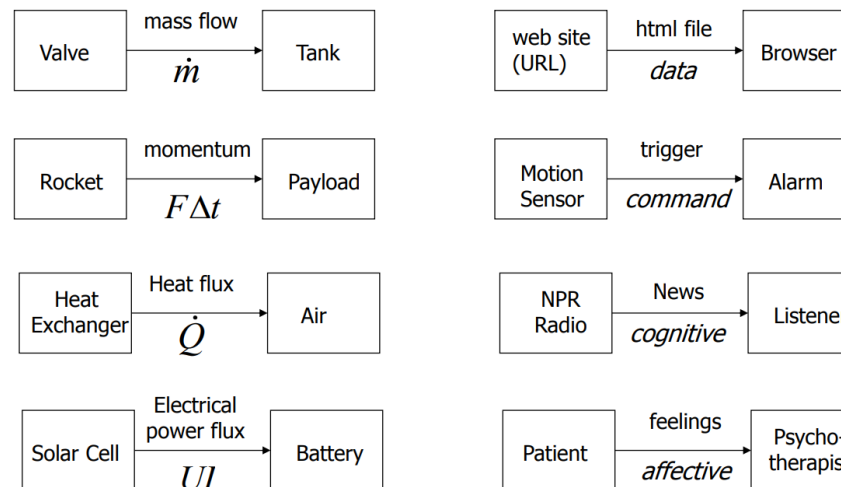
- “Most of time, digital device failed at the interface.” ---- one of my CSEE professor back in the old days.

Next week all the engineering stuff we are going to talk about is about interface:

- Between function and time
- Between logic function and hardware
- Even between hardware connections...

Step 3: The great interface theory

Examples of Interfaces



11

4 Canonical Types of Interfaces

- Physical Connection (always symmetric)
 - If A connects to B, B must also connect to A.
- Energy Flow
- Mass Flow
- Information Flow

12