CSSE2010

CSSE2010 / CSSE7201
Lecture 3

Binary Arithmetic

School of Information Technology and Electrical Engineering
The University of Queensland

Today…

- Admin
- Recap on signed number representations in binary
- Logic gates revisited – from previous lecture
- Binary Arithmetic
- Arithmetic Circuits

- Questions: please put them on to the padlet and I will answer them after the lecture

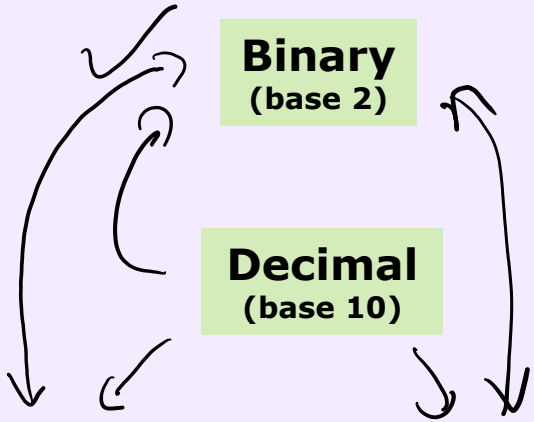
Admin

- Contact eait.mytimetable@uq.edu.au if not yet signed up or you're unable to attend the labs you are signed on to
- Two lab sessions per week from this week. Refer to week-by-week teaching outline on Blackboard.
- All labs are now online due to SEQ lockdown. Zoom information is on Blackboard.
- Weekly exercises (not assessed) are available on Blackboard
- **Quiz 1 – due this week Friday 4pm. Single attempt and need to submit (not auto submitted)**

CSSE2010

Quick Recap – Week 1 Lectures & Learning Lab

Number bases



Binary Formats

2^{N-1}	2^{N-2}	...	2^1	2^0	Excess-2^{N-1} $-2^{N-1} \le x \le (2^{N-1}-1)$
-2^{N-1}	2^{N-2}	...	2^1	2^0	2's comp $-2^{N-1} \le x \le (2^{N-1}-1)$
$-2^{N-1}-1$	2^{N-2}	...	2^1	2^0	1's comp $-(2^{N-1}-1) \le x \le (2^{N-1}-1)$
+/- No mag.	2^{N-2}	...	2^1	2^0	Sign-Mag $-(2^{N-1}-1) \le x \le (2^{N-1}-1)$
2^{N-1}	2^{N-2}	...	2^1	2^0	Unsigned $0 \le x \le 2^N - 1$

MSB N-bit number LSB

Logic Gates

NOT, AND, OR, XOR, NAND, NOR, XNOR
Symbols, Truth table, Boolean expression

Logic diagrams, schematic diagrams
Logic expressions, SOP, Simplification using Boolean algebra

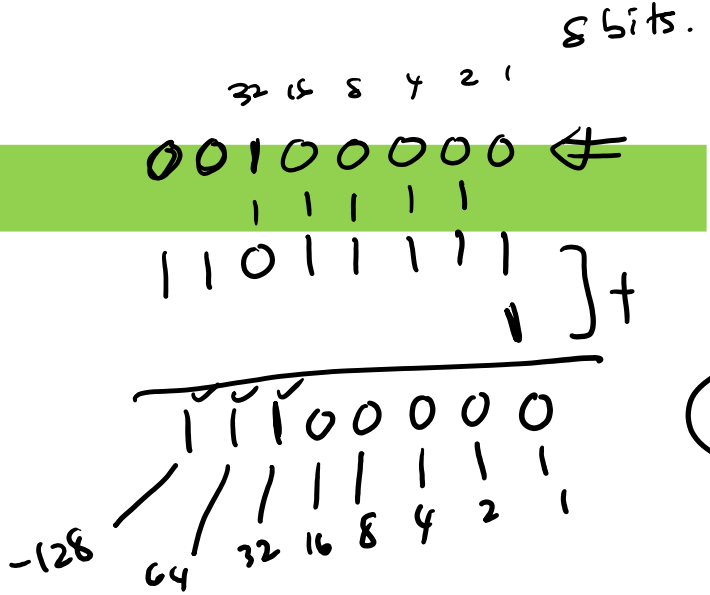
What is -32 (base 10) expressed in 8-bit signed magnitude format?

94%	A. 10100000
1%	B. 10100001
1%	C. 11011111
1%	D. 11100000
1%	E. I don't know

0

What is -32 (base 10) expressed in 8-bit two's complement format?

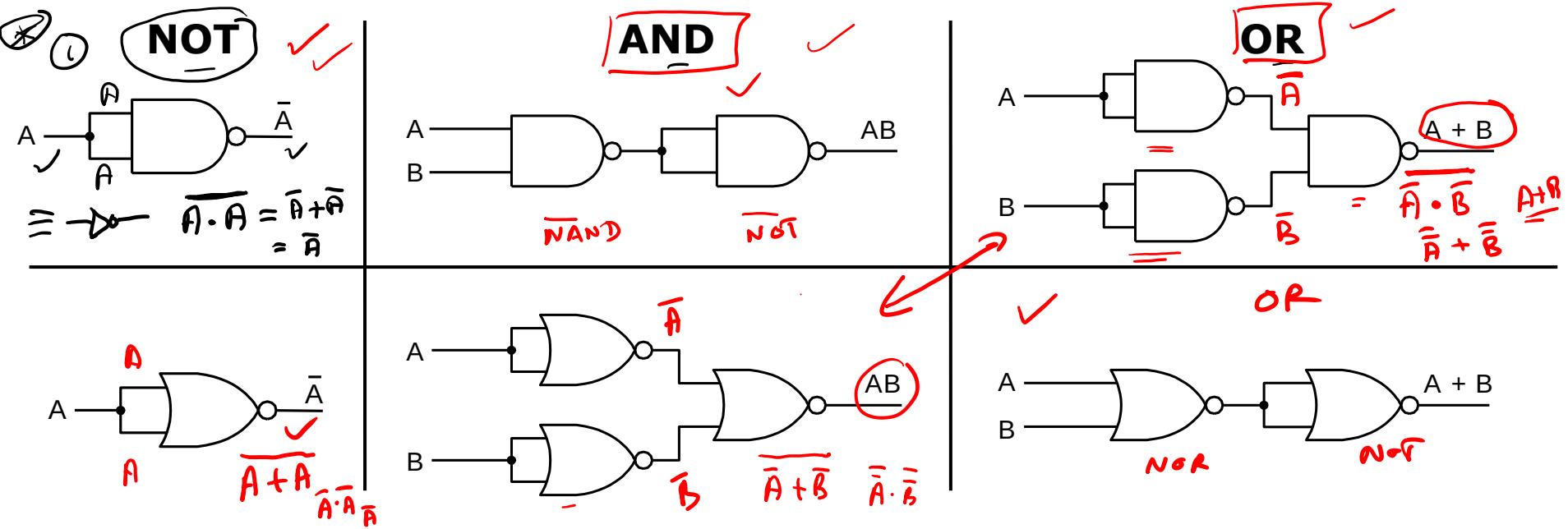
6%	A. 10100000
28%	B. 11011111
8%	C. 10100001
59% ✓✓	D. 11100000 ✓
0%	E. I don't know



0

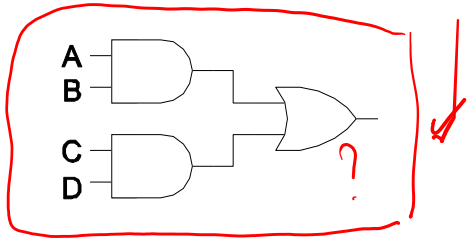
Equivalent Circuits

- All circuits can be constructed from NAND or NOR gates
  - These are called **complete** gates
- Examples:

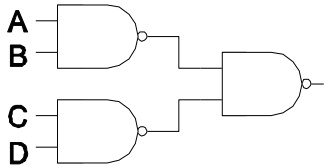


- Reason: Easier to build NAND and NOR gates from transistors

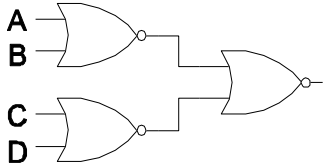
Which of the following is a NOR only implementation of



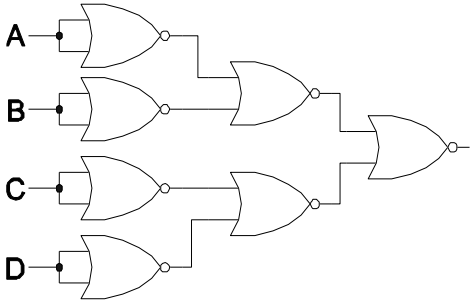
1.



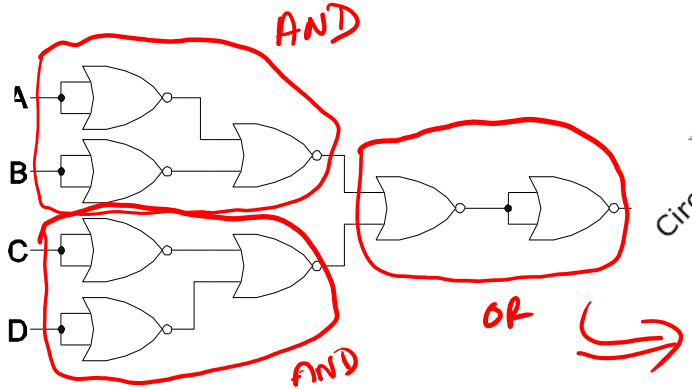
2.

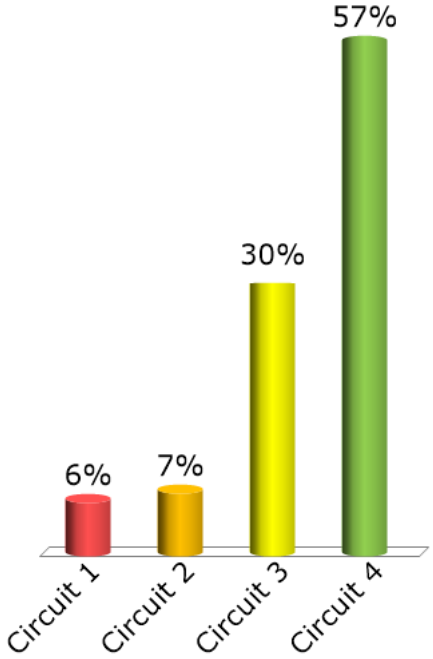


3.



4.





0

Binary Arithmetic

- Addition is quite simple in binary
- We shall see circuits that implement the addition operation later

Addend	0	0	1	1
Augend	+0	+1	+0	+1
Sum	0	1	1	0
Carry	0	0	0	1

- Above ignores carry in

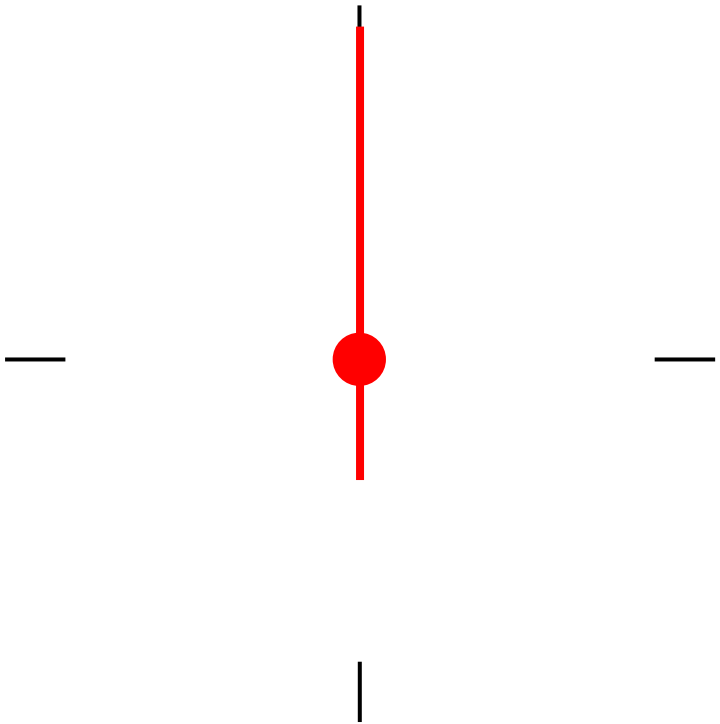
Binary Addition

Decimal	8-bit Unsigned	Decimal	8-bit 2's complement
10	00001010	10	00001010
+ 243	+ 11110011	+(-13)	+ 11110011
253	11111101	-3	11111101

- Format matters when you interpret the numbers
- Whatever the format is the bit-wise addition (which leads to the hardware circuit we will be looking at) is the same
- Two's complement – you don't need to do anything with the carry out from the MSB to get the correct result
- But in one's complement you will have to add the carry out from the MSB back to the result to get the correct answer – this is one drawback of one's complement representation – check by yourself

Short Break

- Stand up and stretch



Overflow in Binary Addition

Decimal	8-bit Unsigned	Decimal	8-bit 2's complement
15	00001111	125	01111101
+ 243	+ 11110011	+ 4	+ 00000100
258	100000010	129	10000001

255

Carry out from MSB

127

-128

125 + 4 ≠ -127

Overflow: Not enough bits to represent the answer. The result goes out of range. Thus, you get a wrong answer.

overflow = $C_{in} \oplus C_{out}$

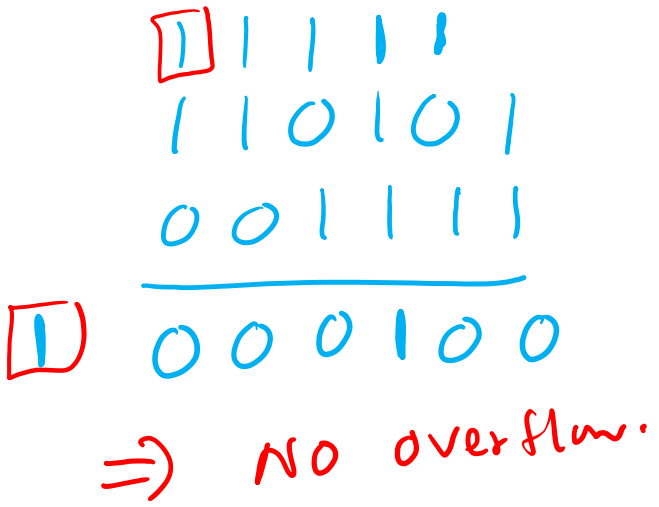
How is overflow detected:

- **Unsigned:** carry-out from the MSB → overflow
- **2's comp:** carry-in to the MSB and carry-out from the MSB are different → overflow
- Equivalently, overflow occurs if (in 2's comp)
  - Two negatives added together give a positive, or
  - Two positives added together give a negative

CSE2010

What is the result of adding the two 6-bit two's complement numbers 110101 and 001111 in 6-bits?

73%	A. 000100
3%	B. 000101
2%	C. 001010
6%	D. 111010
16%	E. 1000100



0

What's the truth table for an adder?

Inputs = A, B

Outputs = S(Sum), C(Carry)

1.

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

2.

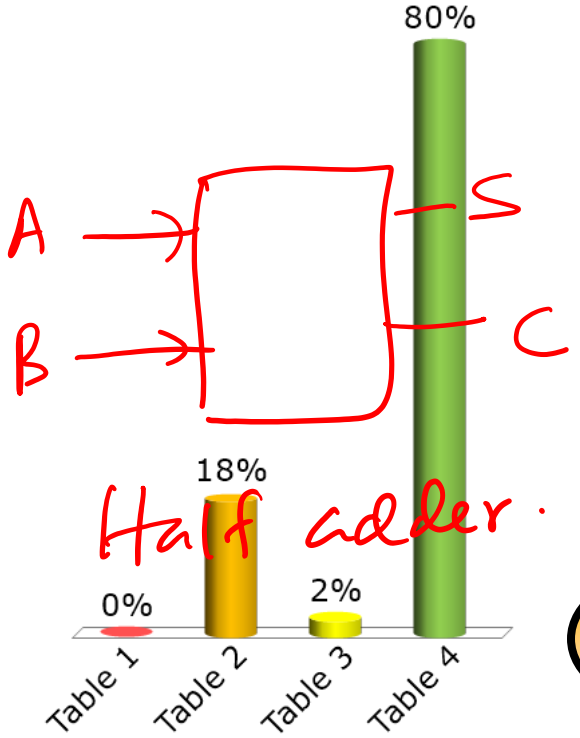
A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

3.

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

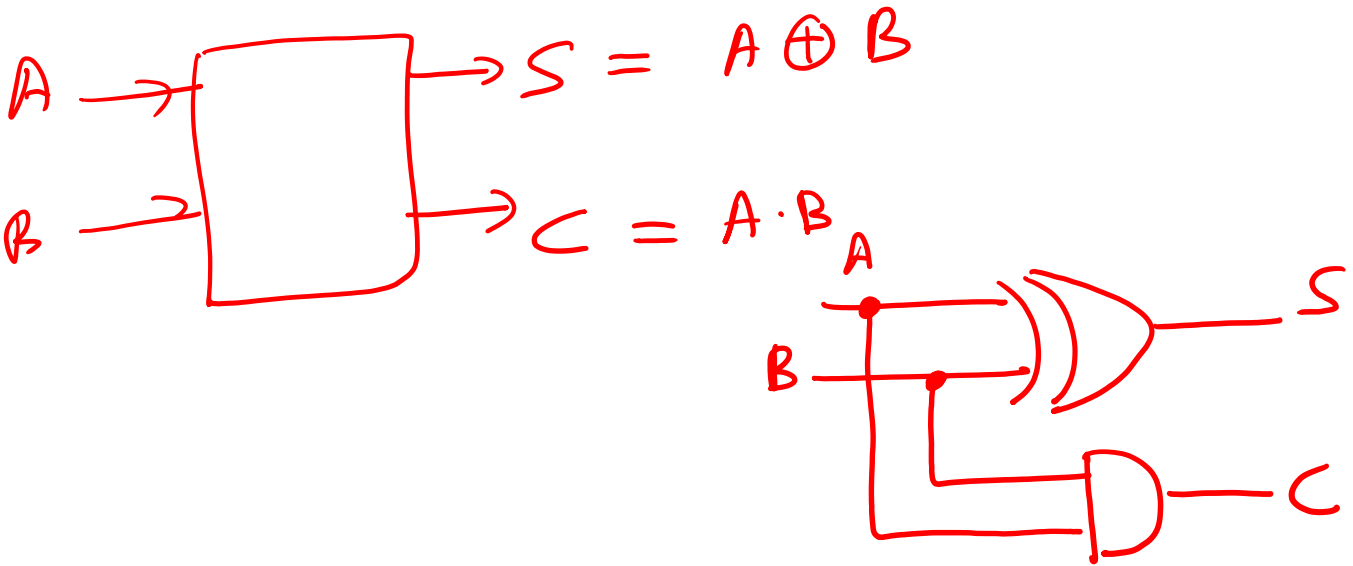
4.

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



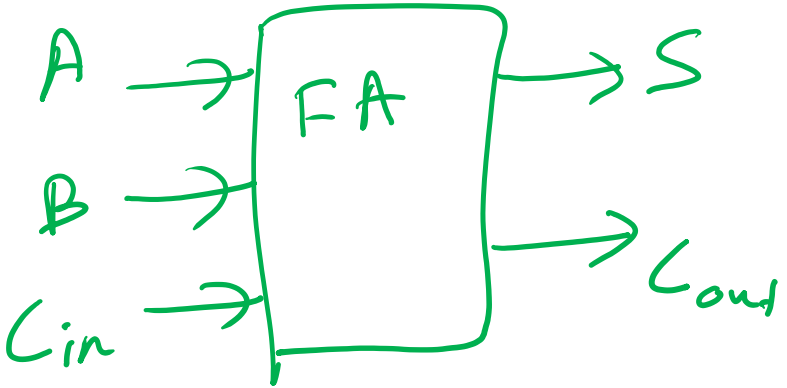
Binary Addition

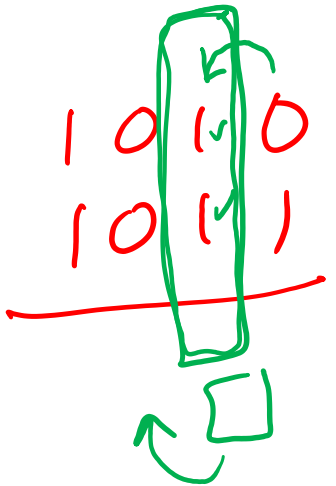
- A device which adds 2 bits (with no carry-in) is called a **half-adder**



Binary Addition

- We have to deal with carry-in. There might be a carry-in from the previous stage.





Addition of Binary Words

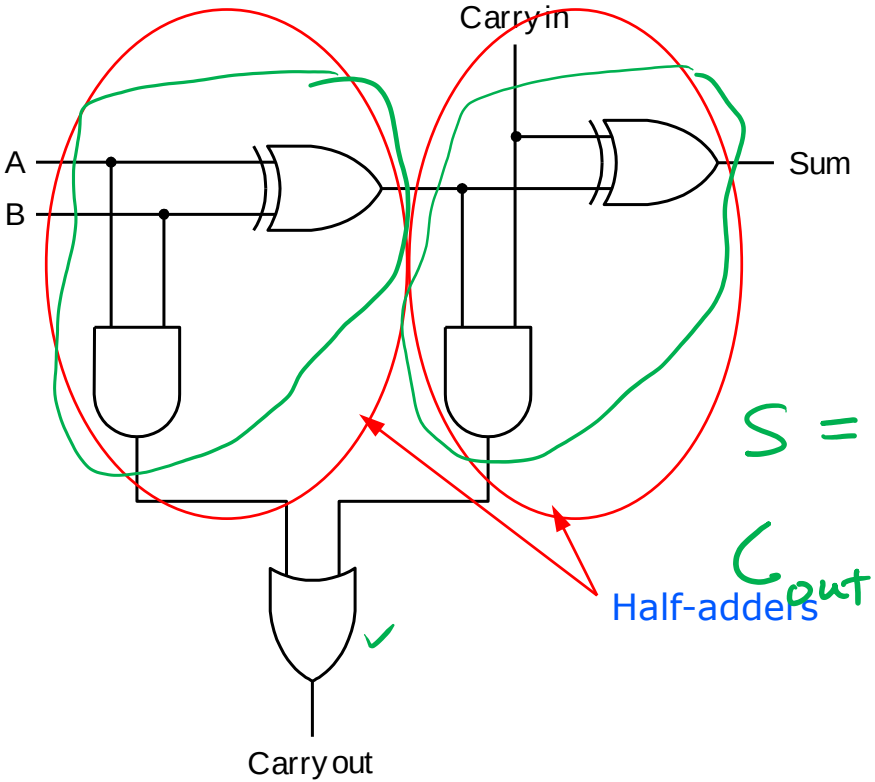
- Have to be able to deal with carry-in
- Truth table to be completed in class

A	B	Cin	Cout	Sum
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

ODD

$A \oplus B \oplus Cin$

Full Adder

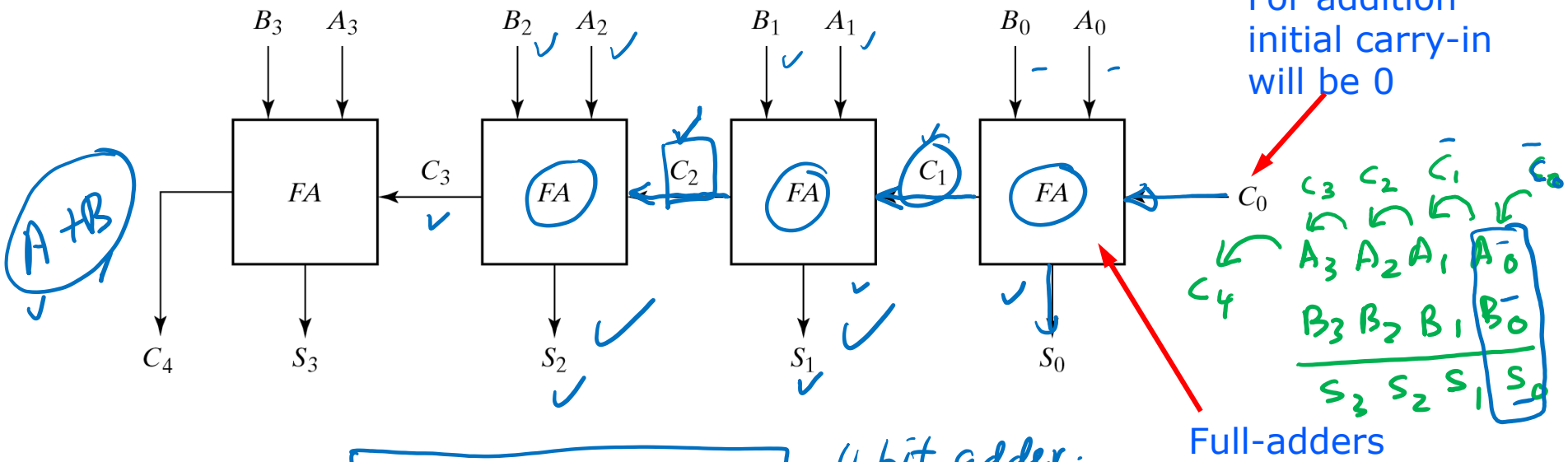


$S = A \oplus B \oplus C_{in}$

$C_{out} = AB + C_{in}(A \oplus B)$

Binary Adder

- Can cascade full adders to make binary adder
  - Example: for 4 bits…



- This is a **ripple-carry adder**

[Figure from Mano, "Digital Design", 3rd edition]

Reminders for this week

- Lab classes
  - Mon-Tue: Lab 2 (Logic Gates)
  - Wed-Fri: Lab 3 (Binary Arithmetic)
- Quiz due this week Friday 4pm
- This week's labs
  - IN students: if you have borrowed kits then use the kits. Otherwise use the Logisim software to simulate the logic circuits.
  - EX students: Use Logisim software to simulate the logic circuits. Start acquiring Arduino based hardware required in week 6.