Intro to Software Engineering
EK327

Giacomo Cappelletto

21/1/25

Contents

Chapter 1

Computer Architecture

1.1 Adding Two Single-Bit Binaries Using XOR and AND Gates

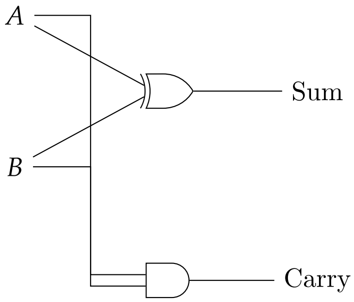
To add two single-bit binary numbers A and B, we need to calculate: 1. The **sum bit**, which is the XOR of A and B. 2. The **carry bit**, which is the AND of A and B.

Logic - Sum Bit: Sum = $A \oplus B$ - **Carry Bit:** Carry = $A \cdot B$

The following truth table summarizes the operation:

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

The addition is implemented using XOR and AND gates as shown below:



The XOR gate produces the sum bit, while the AND gate produces the carry bit. This forms the fundamental building block of a full binary adder.

1.2 Adding Two Single-Byte Binary Numbers Using XOR and AND

Binary addition can be performed on two single-byte numbers using bitwise operations. Specifically: - The **XOR (Exclusive OR)** operation gives the sum of two bits without considering the carry. - The **AND** operation identifies where a carry will occur. - The carry is then shifted left by one position and added to the result in subsequent iterations.

Algorithm: 1. Compute the sum without carry using XOR:

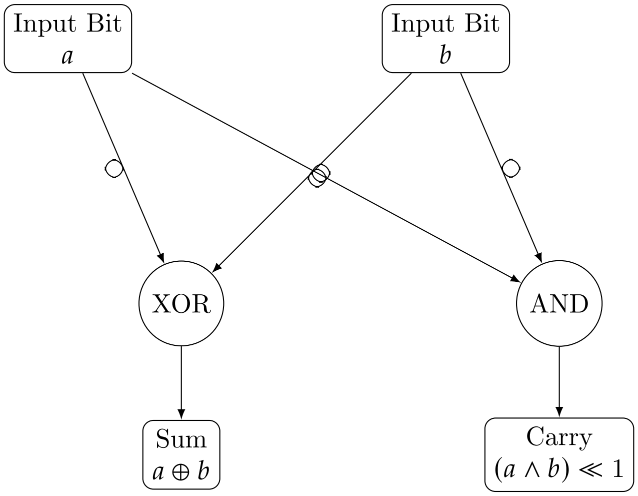
$$\text{sum} = a \oplus b$$

2. Compute the carry using AND and shift it left by one bit:

$$\text{carry} = (a \wedge b) \ll 1$$

3. Repeat the process by adding the carry to the sum until there is no carry left.

Diagram: Below is a diagram that illustrates the process for a single bit addition.



Example: Consider two single-byte binary numbers:

$$a = 01101101, \quad b = 10101001$$

1. Compute the XOR for the sum:

$$\text{sum} = a \oplus b = 11000100$$

2. Compute the AND for the carry and shift left:

$$\text{carry} = (a \wedge b) \ll 1 = 00001010$$

3. Add the sum and carry: - New sum: sum = $11000100 \oplus 00001010 = 11001110$ - New carry: carry = $(11000100 \wedge 00001010) \ll 1 = 00000000$

4. Final result: 11001110.