

Exercise walk-through

3.2

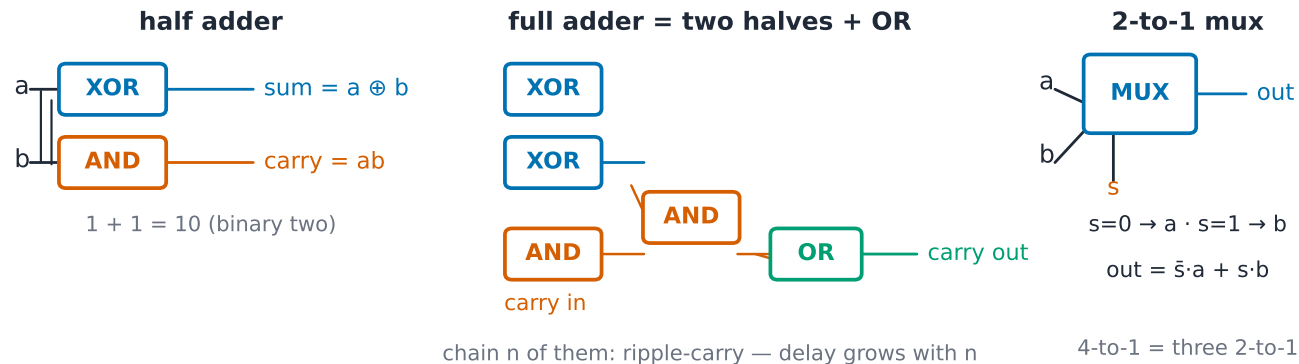
Logic gates and digital building blocks

University Computer Science

Logic gates: from truth tables to adders

- 1 **Half adder** $\text{Sum} = a \text{ XOR } b$, $\text{carry} = ab$ — $1+1 = 10$ in two gates; **full adder** = two halves + OR, chained into ripple-carry (delay grows with n).
- 2 **Mux** $\text{Out} = \bar{s} \cdot a + s \cdot b$: $s=0$ passes a , $s=1$ passes b ; a 4-to-1 is three 2-to-1 stages.
- 3 **Algebra** $x + x = x$, $x + \bar{x} = 1$ — fewer gates, less power, less delay; gate delay limits clock speed.

Gate schematic: half adder, full adder, mux



NAND is universal — every gate above can be built from NAND alone

boolean identities minimise circuits: fewer gates $\rightarrow$ less power, less delay

XOR-and-AND make a **half adder**; two make a **full adder**; a mux picks a lane.

Exercises 2.1-2.3

2.1 Give the sum and carry of $1 + 1$ in a half adder. [3]

2.2 How many 2-to-1 muxes make a 4-to-1? [3]

2.3 Simplify $x \cdot \bar{x}$. [3]

Exercises 2.1-2.3

2.1 Give the sum and carry of $1 + 1$ in a half adder. [3]

2.1

- Sum 0, carry 1 (binary 10) ✓.

2.2 How many 2-to-1 muxes make a 4-to-1? [3]

2.2

- Three ✓.

2.3 Simplify $x \cdot \bar{x}$. [3]

2.3

- 0 – always false ✓.

Exercise 3.1

3.1 Design a 2-bit ripple-carry adder from half and full adders.

Name the pieces. Write each output as a boolean expression. Trace $11_2 + 01_2$ through your circuit and give the sum and the final carry. [8]

Exercise 3.1

3.1 Design a 2-bit ripple-carry adder from half and full adders.

Name the pieces. Write each output as a boolean expression. Trace $11_2 + 01_2$ through your circuit and give the sum and the final carry. [8]

3.1

- Pieces: half adder (bit 0) + full adder (bit 1) ✓.
- $s_0 = a_0 \oplus b_0$; $c_0 = a_0 b_0$ ✓.
- $s_1 = a_1 \oplus b_1 \oplus c_0$; $c_1 = a_1 b_1 + c_0(a_1 \oplus b_1)$ ✓.
- Trace: $a = 11$, $b = 01$: $s_0 = 1 \oplus 1 = 0$, $c_0 = 1$; $s_1 = 1 \oplus 0 \oplus 1 = 0$, $c_1 = 0 + 1 = 1 \rightarrow 100$ ✓.

Exercise 3.2

3.2 Two circuits are claimed to implement the same 2-to-1 mux: $f = \bar{s}a + sb$ and $g = \bar{s}a + sb + s\bar{s}\bar{a}$.

Simplify g , showing every step, and prove algebraically whether the claim holds. Build the joint truth table for both outputs and confirm they match row by row. State which circuit you would build and why. [7]

Exercise 3.2

3.2 Two circuits are claimed to implement the same 2-to-1 mux: $f = \bar{s}a + sb$ and $g = \bar{s}a + sb + s\bar{s}\bar{a}$.

Simplify g , showing every step, and prove algebraically whether the claim holds. Build the joint truth table for both outputs and confirm they match row by row. State which circuit you would build and why. [7]

3.2

- $g = \bar{s}a + sb + \bar{s}\bar{a}s$; the last term has $s\bar{s} = 0$ ✓.
- So $g = \bar{s}a + sb = f$ – one identity collapses it ✓.
- Joint table: $s=0 \rightarrow$ both output a ; $s=1 \rightarrow$ both output b ✓.
- Build f : fewer gates, less delay – the identity is the optimiser ✓.