Ch 4 - Combinational Logic

Sum of Products

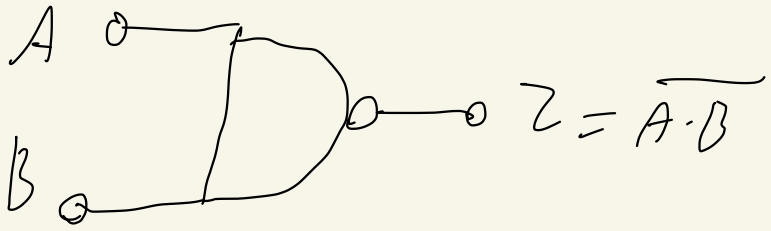
- any function can be built from AND/OR/NOT
  - Inverter: A ⟶ $Z = \overline{A}$
  - AND: A, B ⟶ $Z = A \cdot B$
  - OR: A, B ⟶ $Z = A + B$

Two Approaches for 2 inputs

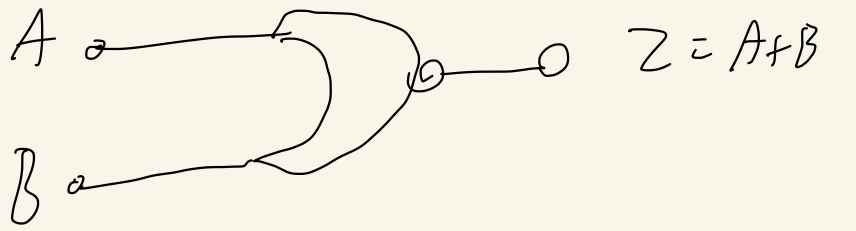
- Chain → propagation delay increased linearly with # of inputs
- Tree → prop. delay increases log. with # of inputs

More Building Blocks

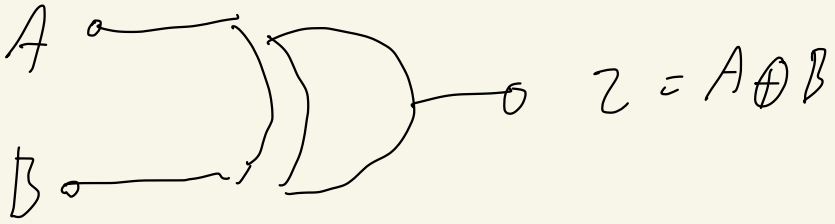
- NAND:



- NOR:



- XOR:



* NAND and NOR are not associative
  - but they are universal
* CMOS love inverting logic

K - Maps

↳ truth table arranged so terms which differ by exactly one variable are adjacent to one another so we can see potential reductions

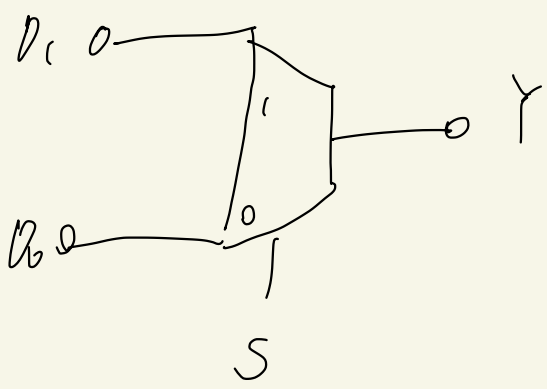
Implicants

↳ rectangular region in K-map has a value 1 and has a width/length of a power of 2

→ **prime implicant** → an implicant that is not completely contained in any other implicant

↳ [large the implicant, the smaller the product term

MUXes

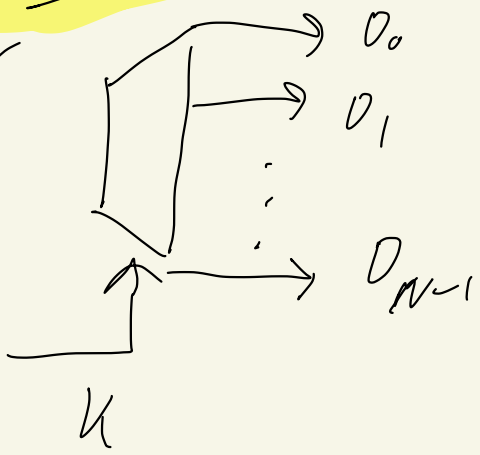


S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

are universal

for an N input function, need a 2^N input MUX

Decoder



- K SELECT inputs
- $N = 2^K$ DATA OUTPUTS
- select inputs chose one of the O_i to assert high, all others to assert low

ROM (Read-only memory)

- 2-d array of decoders and selectors
- model: → look up value of a function in a truth table
  - input: address of a T.T entry
  - ROM size = # of T.T entries
    - for N-input: $2^N \times$ # outputs