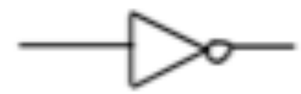
Digital Logic and Computer Organization

Boolean Functions and Circuits

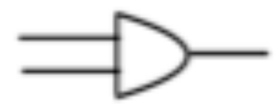
Basic Logic Operations & their gate symbols

- Buffer

- NOT



- AND



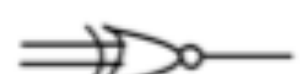
- OR

- NOR

- NAND

- Exclusive-OR (XOR)

- Exclusive-NOR (XNOR)



Boolean Functions

- A logic circuit implements a Boolean function
- Boolean function consists of binary variables, the constants, and the logic operation symbols.
- A Boolean function can be evaluated to 0 or 1 for a given value of the binary variables.
- A Boolean function can be represented as a Boolean algebraic expression, a truth table and/or a schematic diagram.
- One function, one truth table, but multiple equivalent expressions and schematic diagrams.

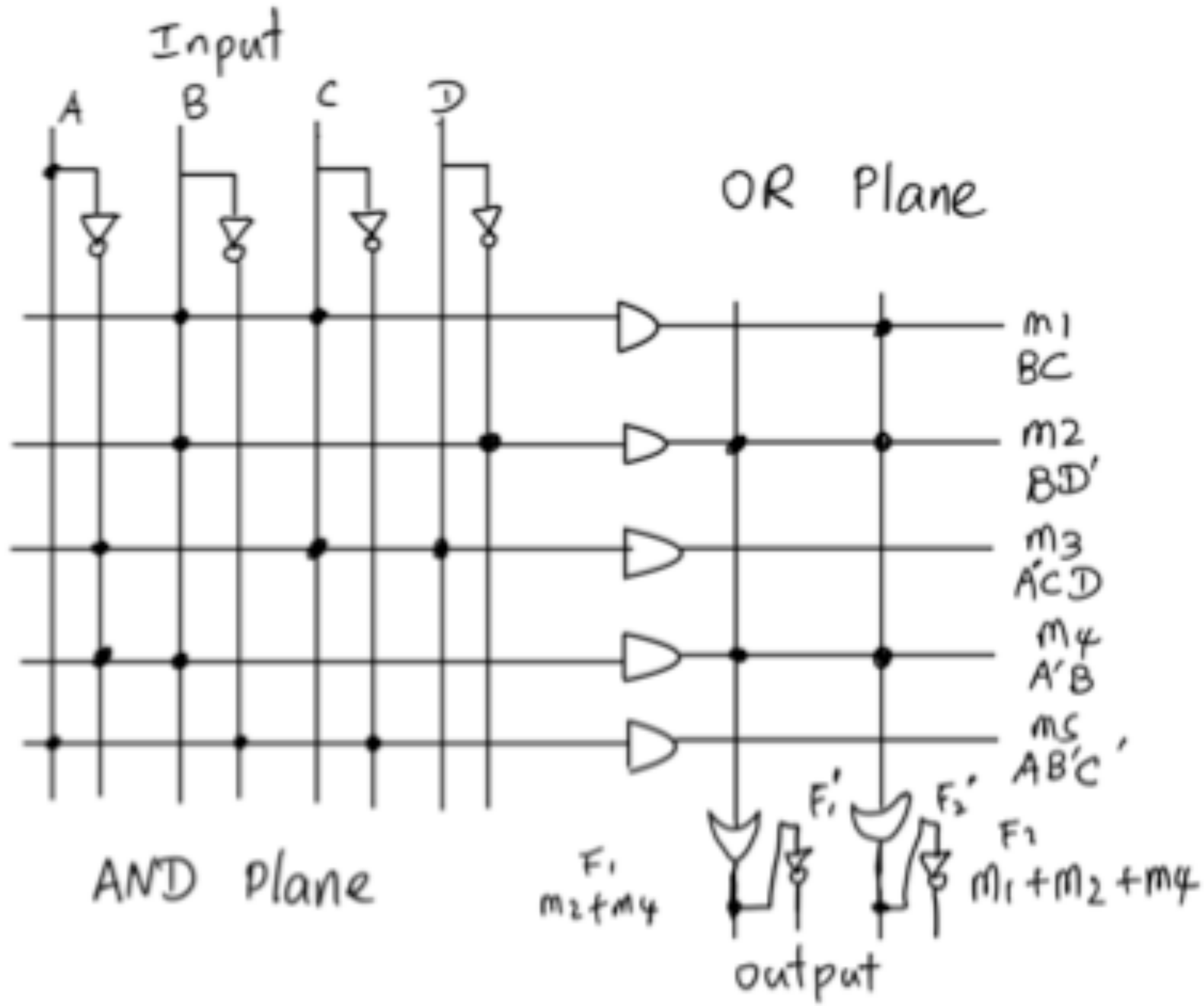
Canonical Forms

- A binary variable may appear either in its normal form (x) or in its complement form (x’).
- A truth table with n variables has 2^n rows
- minterm (standard product) — mi
- maxterm (standard sum) — Mi
- sum of minterms (sum of products)
- product of maxterms (product of sums)
- Minimization - find equivalent expression with minimal number of literals

Minimization Criteria (for SOP)

- Criteria:
  - minimize the total number of product terms
  - Minimize the size of the product terms (minimize the number of literals in each product term)
  - the number of inverters doesn’t matter
- Because the implementation is usually done using Programmable Logic Arrays

Programmable Logic Array (abstract view)



Karnaugh (K) Map

- One function, one truth table, but multiple equivalent expressions.
- Simplification using algebraic method lacks specific rules to guide the manipulative process.
- Map method (Karnaugh map method) provides a simple and straightforward procedure for the minimization process.
- K-Map uses Gray Code ordering
- 2/3/4/5 variable K map examples

K-Map Terminology (for sum of products)

- Implicant: any (power of 2) grouping of adjacent 1’s
- Cover: a set of implicants that include all the 1’s
- Prime Implicant (PI): an implicant that can not be “grown” any bigger
- Essential Prime Implicant: a PI that must be included in a cover
- Secondary Prime Implicant (Non Essential PI): an implicant that is not an essential PI

K-Map Minimization Algorithm (for sum of products)

- Identify all prime implicants;
- Identify the set of essential prime implicants E;
- Select the minimum set of non-essential prime implicants N such that (E union N) forms a cover;

Minimization for Product of sums

- Use DeMorgan's Theorem:
  - minimal POS for f = (minimal SOP for f')'
  - minimal SOP for f = (minimal POS for f')'
- Given the K map for a boolean function
  - find the minimal sum of product for f'
    (e.g., f' = ab + a'cd' + bc'd
  - f = (f')' = (ab + a'cd'+bc'd)' = (ab)'(a'cd')'(bc'd)'
    = (a'+b')(a+c'+d)(b'+c+d')

Don’t Care Condition

- Example
- How do we handle them: don’t care, assign to 0 or 1 depends on how to maximize the benefit of minimization