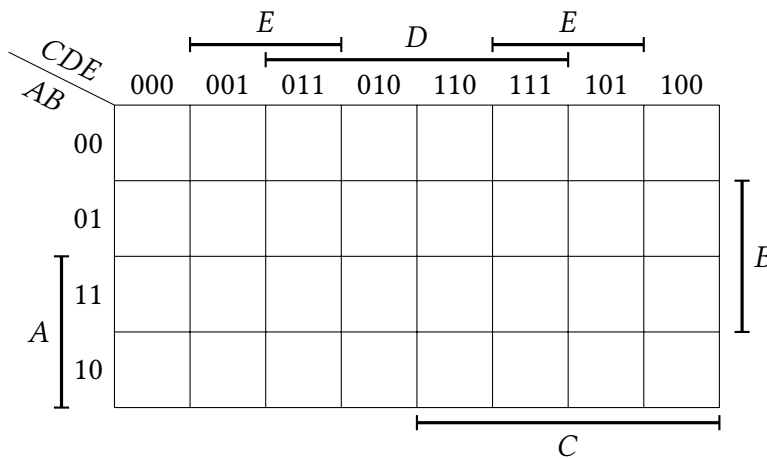


1. Perform the following steps:

- Calculate the SHA-256 hash h of the string $s = \text{"DM Fall 2023 HW3"}$ (without quotes, with all spaces, encoded in UTF-8). Convert hash h to a 256-bit binary string b (prepend leading zeros if necessary). Cut the binary string b into eight 32-bit slices r_1, \dots, r_8 , e.g. $r_2 = b_{33..64}$. Xor all slices into a 32-bit string $d = r_1 \oplus \dots \oplus r_8$. Compute $w = d \oplus 0x24d03294$.
Hint: last (least significant) bits of h are $\dots 01001001$, last bits of d are $\dots 0001$.
- Draw the Karnaugh map (use a template below) for a function $f(A, B, C, D, E)$ defined by the truth table $w = (w_1 \dots w_{32})$, where MSB corresponds to $f(0) = w_1$ and LSB to $f(1) = w_{32}$.
- Use K-map to find the minimal DNF and minimal CNF for the function f .
- Use K-map to find the number of prime implicants, i.e. the size of BCF.



2. For each given function f_i of 4 arguments, draw the Karnaugh map and use it to find BCF, minimal DNF, and minimal CNF. Additionally, construct ANF (Zhegalkin polynomial) using either the K-map, the tabular ("triangle") method or the Pascal method – use each method at least once.

Note: WolframAlpha¹ interprets the query " n -th Boolean function of k variables" in a reverse manner. In order to employ WolframAlpha properly, manually flip the truth table beforehand, e.g. the correct query for $f_{10}^{(2)}$ is "5th Boolean function of 2 variables"², which gives $f_{10}^{(2)} = \neg x_2$, since $\text{rev}(1010_2) = 0101_2 = 5_{10}$.

- $f_1 = f_{47541}^{(4)}$
- $f_2 = \sum m(1, 4, 5, 6, 8, 12, 13)$
- $f_3 = f_{51011}^{(4)} \oplus f_{40389}^{(4)}$
- $f_4 = \overline{A}BD + \overline{A}\overline{C}D + \overline{B}C\overline{D} + A\overline{C}D$

3. Convert the following formulae to CNF.

- $X \leftrightarrow (A \wedge B)$
- $Z \leftrightarrow \bigvee_i C_i$
- $D_1 \oplus \dots \oplus D_n$
- $\text{majority}(X_1, X_2, X_3)$ ¹
- $R \rightarrow (S \rightarrow (T \rightarrow \bigwedge_i F_i))$
- $M \rightarrow (H \leftrightarrow \bigvee_i D_i)$

4. For each given system of functions F_i , determine whether it is functionally complete using Post's criterion. For each basis F_i , use it to represent the majority(A, B, C) function. Draw a combinational Boolean circuit for each resulting formula.

- $F_1 = \{\wedge, \vee, \neg\}$
- $F_2 = \{f_{14}^{(2)}\}$
- $F_3 = \{\rightarrow, \nrightarrow\}$
- $F_4 = \{1, \leftrightarrow, \wedge\}$

5. Show – without using Post's criterion – that the Zhegalkin basis $\{\oplus, \wedge, 1\}$ is functionally complete.

¹ Majority function² is a Boolean function that is 1 iff the majority (more than half) of the inputs are 1.

-
- The diagram shows a logic circuit for a 3-input majority function. It has three inputs: A, B, and C. The circuit consists of several logic gates: AND gates, OR gates, and a final OR gate. The output is labeled $f(1)$.

- Binary Decision Diagram ^{[↗](#)} (BDD) is a representation of a Boolean function as a directed acyclic graph, which consists of *decision* nodes and two *terminal* nodes (0 and 1). Each decision node is labeled by a Boolean variable x_i and has two child nodes called *low* and *high*. The edge from node to a low (high) child represents an assignment of the value FALSE (TRUE) to variable x_i . A path from the root node to the 1-terminal (0-terminal) corresponds to an assignment for which the represented Boolean function is true (false). BDD is called *ordered* if variables appear in the same order on all paths from the root. BDD is called *reduced* if it does not contain a node v with $\text{high}(v) = \text{low}(v)$, and there does not exist a pair of nodes u, v such that the sub-OBDDs rooted in u and v are isomorphic.

- (a) $f_1(x_1, \dots, x_4) = x_1 \oplus x_2 \oplus x_3 \oplus x_4$ (c) $f_3(x_1, \dots, x_4) = \sum m(1, 2, 5, 12, 15)$
 (b) $f_2(x_1, \dots, x_5) = \text{majority}(x_1, \dots, x_5)$ (d) $f_4(x_1, \dots, x_6) = x_1x_4 + x_2x_5 + x_3x_6$