2.1 Digital Logic

Digital logic forms the core of computing and electronics, using number systems, logic levels, logic gates, Boolean algebra, and simplification methods like Karnaugh maps. Number systems in digital logic include decimal, binary, octal, and hexadecimal. The decimal system, used in everyday life, is base 10 with digits from 0 to 9. The binary system, fundamental to digital circuits, operates with only two digits: 0 and 1. Binary numbers can be converted to decimal by multiplying each bit by the corresponding power of 2 and summing the results. Decimal to binary conversion involves successive division by 2, recording remainders in reverse order. The octal system is base 8, using digits 0 to 7, and serves as a shorthand for binary by grouping bits into sets of three. The hexadecimal system, base 16, consists of digits 0 to 9 and letters A to F, where each hexadecimal digit represents four binary bits, making it useful for compact data representation in computing and memory addressing.

Logic levels define voltage ranges representing binary states. In transistor-transistor logic (TTL), a voltage close to 0V represents logic 0, while a voltage close to 5V represents logic 1. Complementary metal-oxide-semiconductor (CMOS) logic has different voltage levels, with low power consumption and higher noise immunity, where logic 0 ranges from 0V to 1.5V and logic 1 ranges from 3.5V to 5V.

Logic gates are the building blocks of digital circuits, each performing a fundamental logical operation. The AND gate outputs a high signal only if all inputs are high, whereas the OR gate outputs high if at least one input is high. The NOT gate, also known as an inverter, reverses the input signal, outputting high when the input is low and vice versa. NAND and NOR gates are universal, meaning they can be used to construct any other logic gate. The NAND gate gives a low output only when all inputs are high, whereas the NOR gate outputs high only when all inputs are low. The XOR gate outputs high when the inputs differ, while the XNOR gate outputs high when inputs are the same.

Boolean algebra provides mathematical rules to simplify logic expressions, making circuits more efficient. The commutative law states that the order of operands does not affect the result, as seen in $A + B = B + A$ and $A \cdot B = B \cdot A$. The associative law allows grouping of terms without changing results, such as $(A + B) + C = A + (B + C)$. The distributive law ensures that multiplication distributes over addition, expressed as $A \cdot (B + C) = A \cdot B + A \cdot C$. De Morgan's theorems help transform logic expressions, where $(A + B)' = A' \cdot B'$ and $(A \cdot B)' = A' + B'$.

The sum-of-products (SOP) method expresses Boolean functions as a sum of minterms, where each term corresponds to an AND combination of input variables that produce a high output. If a truth table lists an output as 1, the corresponding product term is included in the SOP expression. The product-of-sums (POS) method represents a Boolean function as a product of maxterms, where each term consists of OR combinations of input variables that result in a low output. Truth table rows with an output of 0 contribute to the POS expression.

A truth table lists all possible input combinations and their corresponding outputs, helping in circuit design and verification. Karnaugh maps (K-maps) provide a visual method for simplifying Boolean expressions by grouping adjacent 1s into larger blocks, reducing the number of logic gates required. A four-variable K-map consists of a 4x4 grid where adjacent cells differ by only one bit, allowing efficient grouping and minimization of logic expressions.