CHAPTER -2

Encoding Schemes and Number System

2MARK QUESTIONS

Q1: Define the term 'Number System' and explain the significance of different number systems in computing.

Answer:

A number system is a way of expressing numbers using digits and a base. In computing, different number systems, such as binary, octal, decimal, and hexadecimal, are essential for representing and manipulating data in various formats.

Q2: Describe the binary number system. Provide an example of a binary number and explain how to convert it to its decimal equivalent.

Answer:

The binary number system uses only two digits, 0 and 1. For example, the binary number 1101 is converted to decimal as follows: (1 x 2^3) + (1 x 2^2) + (0 x 2^1) + (1 x 2^0) = 8 + 4 + 0 + 1 = 13 (decimal).

Q3: Explain the concept of the octal number system. Illustrate how to convert an octal number to its decimal equivalent.

Answer:

The octal number system uses digits from 0 to 7. For example, the octal number 34 is converted to decimal as follows: (3 x 8^1) + (4 x 8^0) = 24 + 4 = 28 (decimal).

Q4: Discuss the hexadecimal number system. Provide an example of a hexadecimal number and describe the process of converting it to its decimal equivalent.

Answer:

The hexadecimal system uses digits 0-9 and letters A-F to represent values 0-15. For instance, the hexadecimal number 1A is converted to decimal as follows: (1 x 16^1) + (10 x 16^0) = 16 + 10 = 26 (decimal).

Q5: Differentiate between ASCII and Unicode encoding schemes.

Answer:

ASCII (American Standard Code for Information Interchange) is a character encoding scheme using 7 or 8 bits to represent characters, while Unicode is a more comprehensive encoding system that uses 16 or 32 bits, allowing for a broader range of characters and international symbols.

Q6: Explain the purpose of parity bits in binary encoding.

Answer:

Parity bits are used in binary encoding to detect errors in transmitted data. Even parity ensures an even number of 1s in the binary representation, while odd parity ensures an odd number. If the parity doesn't match, an error is detected.

Q 7: Define the term 'bitwise operations' and provide examples of common bitwise operators.

Answer:

Bitwise operations manipulate individual bits in binary numbers. Examples of bitwise operators include AND, OR, XOR, and NOT, which perform operations on corresponding bits of two binary numbers.

Q8: Discuss the concept of 'two's complement' in binary representation. Explain its significance in representing negative numbers.

Answer:

Two's complement is a method for representing negative numbers in binary. It involves inverting all bits and adding 1 to the binary representation of the positive number. This representation simplifies arithmetic operations involving both positive and negative integers.

Q9: Explain the role of the Excess-3 code in binary-coded decimal (BCD) representation.

Answer:

The Excess-3 code is an encoding scheme in BCD where each decimal digit is represented by adding 3 to the corresponding 4-bit binary representation. This helps avoid the use of codes that might cause errors in arithmetic operations.

Q10: Describe the concept of 'Gray Code' and its applications in digital systems.

Answer:

Gray Code is a binary numeral system in which two successive values differ in only one bit position. It finds applications in rotary encoders, where it prevents errors due to misinterpretation of position during transitions, and in digital communications to reduce errors during signal changes.

4MARK QUESTIONS

Q1: Explain the process of converting a decimal number to binary. Provide an example, and discuss the significance of binary representation in computer systems.

Answer:

Converting a decimal number to binary involves successive division by 2, recording remainders in reverse order. For example, converting 13 to binary: ($13 \div 2 = 6$ remainder 1, $6 \div 2 = 3$ remainder 0, $3 \div 2 = 1$ remainder 1, $1 \div 2 = 0$ remainder 1). The binary representation is 1101. Binary is crucial in computers as it forms the basis for digital data representation and processing.

Q2: Discuss the octal number system and its advantages in computer applications. Provide an example of converting a binary number to octal.

Answer:

The octal number system uses base-8, representing digits from 0 to 7. Octal is advantageous in computer applications as it provides a more concise representation than binary. For example, converting the binary number 1101101 to octal: (1101101 = 155 in decimal, and $155 \div 8 = 19$ remainder 3, $19 \div 8 = 2$ remainder 3). The octal representation is 355.

Q3: Explain the hexadecimal number system and its significance in computer programming. Provide an example of converting a binary number to hexadecimal.

Answer:

Hexadecimal uses base-16 with digits 0-9 and A-F. It is commonly used in computer programming for concise representation. For example, converting the binary number 1101101 to hexadecimal: (1101101 = 8D in hexadecimal). Hexadecimal simplifies working with large binary numbers and provides a more human-readable format in programming.

Q4: Compare and contrast ASCII and Unicode character encoding schemes. Discuss the advantages of Unicode over ASCII in handling diverse character sets.

Answer:

ASCII uses 7 or 8 bits to represent characters, primarily for English. Unicode, using 16 or 32 bits, accommodates a broader range of characters and symbols from multiple languages and special symbols. Unicode's advantage lies in its ability to handle diverse character sets, allowing for internationalization and multilingual support in applications.

Q5: Elaborate on the concept of parity in binary encoding. Discuss how even and odd parity are used for error detection. Provide an example.

Answer:

Parity in binary encoding involves adding a bit to ensure an even or odd number of 1s for error detection. Even parity ensures an even count of 1s, while odd parity ensures an odd count. For example, in even parity, for the binary number 1101, a parity bit of 1 would be added to make the total number of 1s even.

Q6: Describe bitwise AND, OR, and XOR operations. Provide examples to illustrate their application in binary manipulation.

Answer:

Bitwise AND, OR, and XOR operate on corresponding bits of two binary numbers. AND sets a bit to 1 only if both corresponding bits are 1. OR sets a bit to 1 if at least one corresponding bit is 1. XOR sets a bit to 1 if the corresponding bits are different. For example, 1010 AND 1100 results in 1000.

Q7: Explain the purpose of two's complement representation in handling negative numbers in binary. Provide an example and discuss its advantages.

Answer:

Two's complement represents negative numbers by inverting all bits and adding 1 to the binary representation of the positive number. For example, in 4-bit representation, -3 is represented as 1101. Two's complement simplifies arithmetic operations, as addition and subtraction can be performed using the same algorithms for both positive and negative integers.

Q 8: Discuss the application of Gray Code in digital systems. Explain how Gray Code minimizes errors during transitions and its significance in rotary encoders.

Answer:

Gray Code is a binary numeral system where two successive values differ in only one bit position, minimizing errors during transitions. It is used in rotary encoders to prevent misinterpretation of position during changes, reducing errors. Gray Code is crucial in digital systems where accurate position sensing is essential.

Q9: Explore the Excess-3 code in Binary-Coded Decimal (BCD) representation. Discuss its encoding process and how it avoids errors in arithmetic operations.

Answer:

The Excess-3 code represents each decimal digit by adding 3 to its 4-bit binary representation in BCD. This avoids the use of codes that might cause errors during arithmetic operations, ensuring accurate representation and processing of decimal numbers in digital systems.

Q10: Discuss the importance of number systems and encoding schemes in the context of computer architecture and data representation. Highlight their role in ensuring efficient data manipulation and communication in computing.

Answer:

Number systems and encoding schemes are fundamental in computer architecture for data representation and manipulation. They facilitate efficient storage, processing, and communication of data. Different number systems provide varying bases for representation, while encoding schemes like ASCII and Unicode enable standardized character representation. These systems collectively ensure accurate and efficient data handling in computing.

7MARK QUESTIONS

Q1: Explain the significance of different number systems in computing, focusing on binary, octal, and hexadecimal. Provide examples and discuss their applications in computer science.

Answer:

Different number systems play a crucial role in computing for representing and manipulating data. Binary is the foundation for digital representation, with applications in machine language and computer architecture. Octal and hexadecimal provide more compact representations, often used in programming and data storage. For example, binary is used in machine language instructions, octal in Unix file permissions, and hexadecimal in memory addresses.

Q2: Describe the process of converting a binary number to its hexadecimal equivalent. Provide step-by-step instructions and illustrate with an example. Discuss the advantages of using hexadecimal in comparison to binary.

Answer:

Converting binary to hexadecimal involves grouping binary digits into sets of four and then converting each set to its hexadecimal equivalent. For example, the binary number 110110101011 can be grouped as 1101 1010 1011 and converted to DAB in hexadecimal. Hexadecimal is more concise than binary, making it easier for humans to read and reducing the length of data representations, especially in programming.

Q 3: Discuss the importance of Unicode in comparison to ASCII for character encoding. Explain how Unicode addresses the limitations of ASCII and its role in supporting multilingual text. Provide examples.

Answer:

Unicode is a more comprehensive character encoding scheme compared to ASCII. ASCII is limited to 128 characters and is primarily focused on English. Unicode, with its larger encoding space, supports a vast array of characters and symbols from multiple languages, making it suitable for internationalization. For example, ASCII cannot represent Chinese characters directly, but Unicode can.

Q4: Explore the concept of parity in binary encoding. Discuss the differences between even parity and odd parity, and explain how parity bits are used for error detection. Provide examples to illustrate.

Answer:

Parity in binary encoding involves adding an extra bit to ensure a certain number of bits are either even or odd. Even parity ensures an even number of set bits, while odd parity ensures an odd number. Parity bits are used for error detection; if the parity doesn't match the expected value, an error is detected. For example, in even parity, for the binary number 1101101, an additional bit of 1 would be added to make the total number of set bits even.

Q5: Elaborate on the applications of Gray Code in digital systems. Discuss its role in minimizing errors during transitions and its significance in devices like rotary encoders. Provide real-world examples.

Answer:

Gray Code is used in digital systems to minimize errors during transitions between binary numbers. In devices like rotary encoders, Gray Code prevents misinterpretation of position during changes, reducing errors. For example, in a rotary encoder representing positions as binary numbers, using Gray Code ensures that only one bit changes at a time during position transitions, minimizing errors.

Q6: Discuss the advantages and disadvantages of using two's complement representation for negative numbers in binary. Explain how two's complement simplifies arithmetic operations and provides examples.

Answer:

Two's complement simplifies arithmetic operations by allowing addition and subtraction to be performed using the same algorithms for both positive and negative integers. The representation is advantageous in digital systems, but it has a disadvantage of having two representations for zero (positive and negative zero). For example, in 4-bit representation, the two's complement of -3 is 1101.

Q7: Explain the application of the Excess-3 code in Binary-Coded Decimal (BCD) representation. Discuss how Excess-3 encoding is used for representing decimal digits and its significance in avoiding errors in arithmetic operations. Provide examples.

Answer:

The Excess-3 code represents each decimal digit by adding 3 to its 4-bit binary representation in BCD. This encoding simplifies arithmetic operations and avoids errors by ensuring that each digit is encoded uniquely. For example, the Excess-3 code for the decimal digit 5 is 1000.

Q8: Discuss the role of different bitwise operators (AND, OR, XOR, NOT) in binary manipulation. Provide examples to illustrate their applications in digital systems. Explain how these operators affect individual bits.

Answer:

Bitwise operators manipulate individual bits in binary numbers. AND sets a bit to 1 only if both corresponding bits are 1. OR sets a bit to 1 if at least one corresponding bit is 1. XOR sets a bit to 1 if the corresponding bits are different. NOT flips the value of a bit. For example, 1010 AND 1100 results in 1000.

Q9: Elaborate on the concept of the encoding scheme used in Binary-Coded Decimal (BCD). Discuss its advantages and applications in digital systems. Provide examples to illustrate BCD representation.

Answer:

BCD is an encoding scheme where each decimal digit is represented by its 4-bit binary equivalent. This simplifies arithmetic operations on decimal numbers in digital systems. BCD is advantageous in applications where precise decimal representation is required, such as financial calculations. For example, the BCD representation of the decimal number 25 is 0010 0101.

Q10: Explore the use of various number systems and encoding schemes in computer architecture. Discuss their role in data representation, storage, and processing. Provide examples of their applications in modern computer systems.

Answer:

Number systems and encoding schemes are fundamental in computer architecture. Binary is used for machine-level instructions, while hexadecimal and octal provide compact representations in programming. ASCII and Unicode are crucial for character encoding in text. Parity bits help in error detection, Gray Code minimizes errors during transitions, and BCD simplifies decimal arithmetic. Their collective role ensures accurate and efficient data representation, storage, and processing in modern computer systems.

Multiple-Choice Questions (MCQs):

Q 1: What is the base of the binary number system?

a) Base-8

b) Base-10

c) Base-16

d) Base-2

Answer: d) Base-2

Q 2: In the hexadecimal number system, what does the symbol 'A' represent?

a) 10

b) 11

c) 12

d) 15

Answer: a) 10

Q 3: What is the purpose of parity bits in binary encoding?

a) Represent negative numbers

b) Detect errors

c) Indicate even numbers

d) Convert to hexadecimal

Answer: b) Detect errors

Q 4: Which bitwise operation sets a bit to 1 if at least one corresponding bit is 1?

a) AND

b) OR

c) XOR

d) NOT

Answer: b) OR

Q 5. What is the significance of Gray Code in digital systems?

a) Minimizes errors during transitions

b) Represents negative numbers

c) Converts to decimal easily

d) Used in ASCII encoding

Answer: a) Minimizes errors during transitions

Q 6: In Unicode encoding, how many bits are typically used for character representation?

a) 8

b) 16

c) 32

d) 64

Answer: b) 16

Q 7: What is the primary advantage of using two's complement representation for negative numbers?

a) Compact storage

b) Easy conversion to hexadecimal

c) Simplifies arithmetic operations

d) Minimizes errors

Answer: c) Simplifies arithmetic operations

Q 8: What does the Excess-3 code represent in Binary-Coded Decimal (BCD)?

a) Even numbers

b) Decimal digits

c) Negative numbers

d) Odd numbers

Answer: b) Decimal digits

Q 9: What is the primary application of BCD (Binary-Coded Decimal) representation?

a) Text encoding

b) Decimal arithmetic

c) Error detection

d) Minimizing transitions

Answer: b) Decimal arithmetic

Q 10: Which number system is commonly used for machine-level instructions in computer architecture?

a) Octal

b) Binary

c) Hexadecimal

d) Decimal

Answer: b) Binary

Fill in the Blanks:

Q1: The hexadecimal system uses digits from __ to __.

Answer: 0 to F

Q 2: Parity bits are used for error __ in binary encoding.

Answer: Detection

Q3: Bitwise __ operation sets a bit to 1 only if both corresponding bits are 1.

Answer: AND

Q4: Gray Code is used to minimize errors during __ between binary numbers.

Answer: Transitions

Q5: Unicode typically uses __ bits for character representation.

Answer: 16

Q6: Two's complement representation simplifies arithmetic __ for both positive and negative integers.

Answer: Operations

Q7: The Excess-3 code represents each decimal digit by adding __ to its 4-bit binary representation in BCD.

Answer: 3

Q8: BCD (Binary-Coded Decimal) is used for precise decimal __ in digital systems.

Answer: Arithmetic

Q9: In binary encoding, parity bits are used to detect __ during data transmission.

Answer: Errors

Q10: The primary application of Gray Code is to minimize errors in devices like rotary __.

Answer: Encoders