Shobhit

Institute of Engineering & Technology

Deemed to-be-University

EDUCATION EMPOWERS

Self-Learning Material

Open and Distance Learning Program

MASTER OF COMPUTER APPLICATION (MCA)

OLMCA005 – COMPUTER ORGANIZATION AND ARCHITECTURE

Prepared by: Dr. Mamta Bansal

ACCREDITED WITH GRADE A NAAC

Shobhit Institute of Engineering & Technology, Meerut

[NAAC "A' Grade Deemed -to-be University established u/s 3 of UGC Act]

NH-58, Modipuram, Meerut - 250110, NCR Delhi, INDIA

[illegible]

Shobhit Institute of Engg. & Tech.

(Deemed to-be-University)

NH-58, Modipuram, Meerut-[illegible]110

COMPUTER ORGANIZATION AND ARCHITECTURE

UNIT – 1 BINARY NUMBERS AND ARITHMETIC

STRUCTURE

1.0 OBJECTIVES

At the end of this unit, the student will be able to:

- Explain the number system and its usage in computer systems
- Describe the conversion of number systems from one to the other
- Discuss the Binary-complement of number systems
- Explain the BCD and ASCII number systems
- Analyze the concept of weighted and unweighted codes
- Apply the Binary arithmetic and Computer Arithmetic
- Analyze the concept of Logic gates and k-map
- Describe the fundamentals of Boolean algebra

1.1 INTRODUCTION

Decimal number system (base 10) is the standard number system followed for counting and measurements in daily use. Computers, being made from transistors use binary (base 2) number system. They operate in two states - on and off. Another widely used number system in computing, is hexadecimal (base 16) or octal (base 8) number systems. They follow a compact form for representing binary numbers. In general, a number system of base, or radix, r uses r digits with distinct symbols. In any number system, numbers are represented by a string of valid digit symbols of that system. To determine the quantity the number represents, it is necessary to multiply each digit by the weight of the digit, which is an integer power of r and then form the sum of all weighted digits.

1.2 DATA AND THE NUMBER SYSTEM

The term data refers to factual information used for analysis or reasoning. These _raw facts' are processed to arrive at meaningful information. Data stored in digital computers are binary-coded. Information is a processed data or computational results that is communicated. Digital computers store binary information in memory or processor registers. These data are either data or control information. Control information indicate the command signals needed for the control of data. These signals may be a bit or group of bits.

Following are the categories of data handled by digital computers:

- Numbers used in calculations
- Letters used in data handling, and
- Other characters used for specific functions

Registers are made up of two-state devices called flip-flops. They can store only 1's and 0's. So, the binary number system is the most ideal system to use in a digital computer. But human being is comfortable doing computations in the decimal number system.

Decimal (Base 10) Number System

Decimal number system has digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. It follows positional notation, wherein, the least-significant digit (right-most digit) is of the order of 10^0 (units or ones), the second right-most digit is of the order 10^1 (tens), the third right-most digit is of the order 10^2 (hundreds), and so on, where ^ denotes exponent. For example, the quantity represented by sequence of digits 322.1 is calculated as:

$3 \times 10^2 + 2 \times 10^1 + 2 \times 10^0 + 1 \times 10^{-1}$

i.e., 3 hundred, plus 2 tens, plus 2 units, plus 1 tenths. This way one can calculate the quantity represented by any decimal number.

Binary (Base 2) Number System

Binary is a base-2 or radix 2 number system that uses two states 0 and 1 to represent a number. The only digits used here are 0 and 1, which are also called as a true state and a false state. Thesedigits are called bits. A sequence of eight bits is called a byte ($8=2^3$). Any number can be represented by these two digits.

Binary number system also follows a positional notation. For example, the string of digits 101001 represents the decimal quantity:

1 x 2^5 + 0 x 2^4 + 1 x 2^3 + 0 x 2^2 + 0 x 2^1 + 1 x 2^0 = 41

To recognize different number systems, the radix or base of the number is included as a subscript. For example, following statement equates binary and decimal forty-one:

$(101001)_2 = (41)_{10}$

Binary operations such as add, subtract, multiply, and divide binary numbers is an essential part of digital systems.

1.3 CONVERSION OF NUMBERS FROM ONE NUMBERSYSTEM TO THE OTHER

Decimal to Binary conversion

Following example shows steps involved in converting $(28)_{10}$ to binary:

Base	Number	Carry
2	28	**0**
2	14	**0**
2	7	**1**
2	3	**1**
	1	

$28_{10} = (11100)_2$

Binary Addition

When we perform additions in any number system, there will be two outputs: Sum (S) and Carry(C).

Rules for binary addition:

Input A	Input B	Sum (S) = A+B	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0

1	1	0	1

Table 1.1 Binary addition Rules for binary subtraction

When we perform subtractions in any number system, there will be two outputs: Subtract (S) andBorrow (B). Here a Borrow 1 is required only when we subtract 1 from 0. So, the result became 0.

Input A, minuend	**Input B, Subtrahend**	**Subtract (S) A-B**	**Borrow (B)**
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

Table 1.2 Binary subtraction Rules for binary multiplication

Input A	**Input B**	**Multiply (M) A x B**
0	0	0
0	1	0
1	0	0
1	1	1

Table 1.3 Binary multiplication

When all the inputs are 1, result is 1 and result is always 0, whenever at least one input is 0.

Rules for binary division:

Four components in any division are Dividend, Divisor, Quotient, and Remainder.

Input A, Dividend	Input B, Divisor	Divide (D) A/B, Quotient
0	0	Not defined
0	1	0
1	0	Not defined
1	1	1

Table 1.4 Binary division

The result is always not defined, whenever the divisor is 0. **Octal (base 8) and Hexadecimal (base 16) Number System**Digits of the octal number system are 0, 1, 2, 3, 4, 5, 6, and 7.

Digits of Hexadecimal number system are: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F, alsocalled as hex digits. Both follow positional notation.

Examples:

Conversion of octal 136.6 to decimal:

$(136.6)_8 = 1 \times 8^2 + 3 \times 8^1 + 6 \times 8^0 + 6 \times 8^{-1}$

$= 1 \times 64 + 3 \times 8 + 6 \times 1 + 6/8 = (94.75)_{10}$

Conversion of hexadecimal F9 to decimal

$(F9)_{16} = F \times 16 + 9 = 15 \times 16 + 9 = (249)_{10}$

An octal digit relates to three binary digits as $8=2^3$ and each hexadecimal digit relates to fourbinary digits as $16= 2^4$.

Convert 261(base 10) to hexadecimal (base 16):

	quotient	remainder
16	261	5
16	16	0
	1	

261D = 105H

1.4 REPRESENTATION 1'S AND 2'S COMPLEMENT

Number theorists have proven that one decimal number can be subtracted from another byfinding the diminished radix complement of the subtrahend or casting out 9s as below:

1. Find the difference of the subtrahend from all nines

2. Add the result to minuend

3. Add the carry if any, back to the result.

This method is also called as taking the nine's complement of the subtrahend. Consider an example to find 167 − 52.

1. 999-52= 947.
2. 167+947=1114

Now, the -carry‖ from the thousands column is added back to the units' place, 1+114=115 Thus, giving us a correct result, i.e., 167 − 52 = 115.

To simplify computer arithmetic, this method has been extended to binary operations too.Complement systems give us advantage over signed magnitude as there is no need to process sign bits separately. Just by looking at its high-order bit, we can check the sign of a number.

One's Complement of a Signed Binary Number

Negative binary numbers in a signed binary number system are represented using One's Complement or 1's Complement method. In one's complement, positive numbers (also known as non-complements) remain unchanged with the sign-magnitude numbers. Positive number alwaysstarts with a _0'. Negative numbers are represented by negating or by taking inverse (taking the one's complement) of its unsigned positive number. Negative numbers in complement form always starts with a -1‖ .

The one's complement of -1‖ is –0‖ and vice versa. For example, just by complementing each bit of 10110101_2, we get its one's complement 01001010_2.

Subtraction of Two Binary Numbers

Consider subtracting two numbers 99 and 23, using one's complement. In decimal this would be:99 – 23 =76.

Step 1: Convert the two decimal numbers into binary. Make sure that each number has the samenumber of bits by adding leading zeros to produce an 8-bit number (byte). Therefore:

$(99)_{10}$ in binary is: $(01100011)_2$

$(23)_{10}$ in binary is: $(00010111)_2$

Step 2: Now find the complement of the subtrahend, by changing all the 1's to 0's and 0's to 1's, the one's complement of 0010111 is equal to 11101000. Adding the minuend and the complement of the subtrahend gives:

01100011+ 11101000

Overflow → 1 01001011

There is an overflow due to the carry forward generated by the sum of most significant column. Positive results will always have an overflow. If the answer is negative, there is no overflow. Theoverflow can be ignored or passed to next circuit if system works with 8 bits.

Now convert the 8-bit answer from a one's complement answer to the real answer by adding -1‖, therefore:

01001011

+ 1

01001100_2

So, the result of subtracting 23 (00010111_2) from 99(01100011_2), using 1's complement in binary gives the answer of: 01001100_2 or (64 + 8+ 4) = 76_{10} in decimal.

Thus, One's Complement has the advantage of subtracting two binary numbers just by using addition. This can be compared to decimal subtraction equivalents such as A – B, is the same as saying A + (-B) or -B + A etc.

Two's Complement of a Signed Binary Number

Positive numbers in two's complement are same as 1's complement. and a negative number is itsone's complement plus one.

Following are advantages of two's complement over one's complement:

1. There is no double-zero problem(1's complement has two representations for zero, positive and negative)

2. It is very easy to generate the two's complement of a signed binary number which results insimpler arithmetic operations.

Consider the subtraction of two 8-bit numbers 99 and 23 using two's complement.

Step 1: Convert the two decimal numbers into binary. Make sure that each number has the samenumber of bits by adding leading zeros to produce an 8-bit number (byte). Therefore: $(99)_{10}$ in binary is: $(01100011)_2$

$(23)_{10}$ in binary is: $(00010111)_2$

Now find the 2's complement of $(23)_{10}$

i.e., $(11101000\ )_2$+1 (complement individual bits +1)=$(11101001)_2$

The complementation of the subtrahend means that the subtraction becomes a much easieraddition of the two numbers, the sum is: 99 + (2's complement of 23) which is: $(01100011)_2$

$+ (11101001)_2 = (101001100)_2$

The 9th overflow bit is disregarded, so the result is: $(01001100)_2$ or $(64 + 8 + 4) = 76_{10}$.

(r-1)'s & r's complement

Digital computers use complement number system for logical operations and for easing the subtraction operation. If we generalize, in any base r system, there are two types of complements:the r's complement and the (r - 1)'s complement. In Binary number system, base r is substituted by 2 and the complement systems are 2's and 1's complement and in decimal system it is 10's and 9's complement.

(r - 1)'s Complement

Given a number N in base r having n digits, the (r - 1)'s complement of N is defined as $(r^n - 1) -$

N. For decimal numbers r = 10 and r - 1 = 9, so the 9's complement of N is $(10^n - 1)$ -N. Now,10^n represents a number that consists of a single 1 followed by n 0's. $10^n - 1$ is a number represented by n 9's. For example, with n = 3 we have $10^3 = 1000$ and $10^3 - 1$

= 999. Thus, by subtracting each digit of a decimal number from 9 results in 9's complement. For example, the 9's complement of 340700 is 999999 - 340700 = 659299 and the 9's complement of 72380 is 99999 - 72380 = 27619.

For binary numbers, r = 2 and r - 1 = 1, so the 1's complement of N is $(2^n - 1) - N$. Again, 2^n is represented by a binary number that consists of a 1 followed by n 0's. $2^n - 1$ is a binary number represented by n 1's. For example, with n = 3, we have $2^3 = (1000)_2$ and $2^3 - 1 = (111)_2$. Thus, bysubtracting each binary digit from 1, 1's complement of a binary number is obtained.

It is also observed that the subtraction of a binary digit from 1 causes the bit to change from 0 to1 and vice versa. Therefore, the 1's complement of a binary number is formed by changing 1's into 0's and 0's into 1's. For example, the 1's complement of 10101010 is 01010101.

These methods of finding (r-1)'s complement holds good in octal or hexadecimal numbersystems too.

(r's) Complement

The r's complement of a n-digit number N in base r is defined as $r^n - N$ for $N \neq 0$ and 0 for N = 0. Comparing with the (r - 1)'s complement, we note that the r's complement is obtained by adding1 to the (r - 1)'s complement since $r^n - N = [(r^n - 1) - N] + 1$. Thus the 10's complement of the decimal 3389 is 6610 + 1 = 6611 and is obtained by adding 1 to the 9's complement value.

The 2's complement of binary 101010 is 010101 + 1 = 010110 and is obtained by adding 1 to the1's complement value.

Following method is one more interesting way to find r's complement. 10's complement of number N, can be formed by leaving all least significant 0's unchanged, subtracting the first nonzero least significant digit from 10, and then subtracting all higher significant digits from 9. The 10's complement of 4680 is 5320 and is obtained by leaving the first zero unchanged, subtracting 8 from 10, and subtracting the other two digits from 9. Similarly, the 2's complementcan be formed by leaving all least significant 0's and the first 1 unchanged, and then replacing 1'sby 0's and 0's by 1's in all other higher, significant bits. The 2's complement of 10010 is 01110 and is obtained by leaving the first 0 and the first 1 unchanged, and then replacing 1's by 0's and 0's by 1's in the other three most significant bits.

1.5 WEIGHTED AND UNWEIGHTED CODES

Digital devices always store and transmit data as groups of binary digits, also known as binarycode. There are multiple categories of binary code:

Weighted codes

In weighted codes, each digit is assigned a specific weight according to its position. Individual bits are multiplied by their positional weights and the sum of these products gives the decimal digit. For example, 1101 in 8421BCD code, the weights of digits from left to right, i.e., 1 0 1 1 are 8, 4, 2 and 1, respectively. The quantity is thus calculated as 1*8+0*4+1*2+1*1=11.

Suppose W1, W2, W3 and W4 are the weights of binary digits and Xl, X2, X3 and X4 are the corresponding digit values then decimal digit, N = W1 Xl + W2 X2 + W3 X3 + W4 X4 is represented by binary sequence X4 X3 X2 X1.

In weighted codes, each digit is assigned a specific weight according to its position. Individual bits are multiplied by their positional weights and the sum of these products gives the decimal digit. For example, 1101 in 8421BCD code, the weights of digits from left to right, i.e., 1 0 1 1 are 8, 4, 2 and 1, respectively. The quantity is thus calculated as 1*8+0*4+1*2+1*1=11.

Suppose W1, W2, W3 and W4 are the weights of binary digits and Xl, X2, X3 and X4 are the corresponding digit values then decimal digit, N = W1 Xl + W2 X2 + W3 X3 + W4 X4 is represented by binary sequence X4 X3 X2 X1.

Non-weighted codes:

The non-weighted codes are not positionally weighted, each digit position within the number is not assigned a fixed value (or weight).

Examples: Excess-3 and Gray code

Reflective codes

A code is reflective when the code is self-complementing. In other words, when the code for 9 is the complement for the code for 0, 8 for 1, 7 for 2, 6 for 3 and 5 for 4.

Example: 2421BCD, 5421BCD and Excess-3 code.

Sequential codes:

In sequential codes, each code is one binary number greater than its preceding code.Example:8421 BCD and Excess-3.

Alphanumeric codes:

Codes used to represent numbers, alphabetic characters, symbols, and various instructions necessary for conveying intelligible information.

Example: ASCII, EBCDIC, UNICODE.

Error defecting and correcting codes:

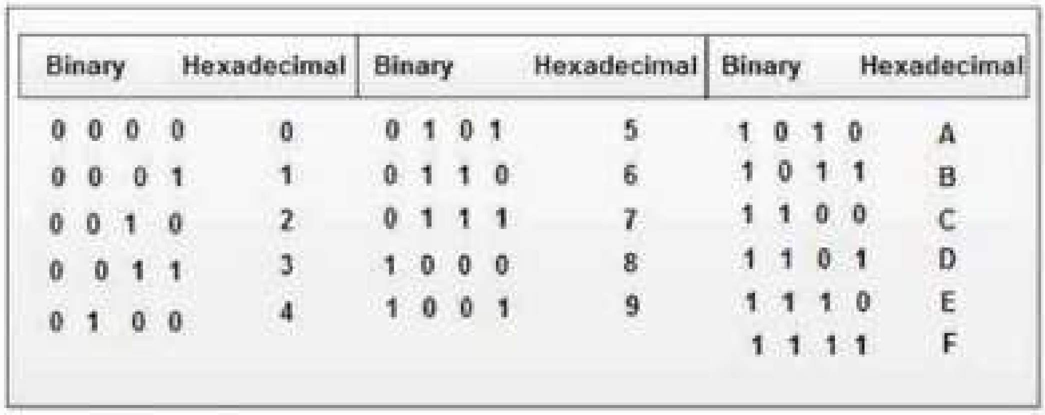
Codes which help in error detection and correction of data are called error detecting and correcting codes.

Example: Hamming code.

Gray Code

Many physical systems supply analog or continuous output data. It must be converted into digitalform to be processed by a digital computer. This conversion is done by analog-to-digital converter which uses Gray code.

The Figure 1.1 lists the Gray code equivalents of the decimal number 0 – 15.



Binary	Hexadecimal	Binary	Hexadecimal	Binary	Hexadecimal
0 0 0 0	0	0 1 0 1	5	1 0 1 0	A
0 0 0 1	1	0 1 1 0	6	1 0 1 1	B
0 0 1 0	2	0 1 1 1	7	1 1 0 0	C
0 0 1 1	3	1 0 0 0	8	1 1 0 1	D
0 1 0 0	4	1 0 0 1	9	1 1 1 0	E
				1 1 1 1	F

Figure 1.1 Gray codes

They are also called as the minimum change code as the successive coded characters never differin more than one-bit. Due to this feature, the maximum error that can creep into a system is muchless than the worst -case error encountered in case of straight binary encoding.

Gray code is not suitable for arithmetic operations as it is an unweighted code. They are also used in input/output devices, and Karnaugh map etc.

A three-bit Gray code can be obtained by merely reflecting the two-bit code about an axis at the end of the code and assigning a third bit as 0 above the axis and as 1 below the axis. The reflected Gray code is nothing but code written in reverse order. By reflecting three-bit code, a four-bit code may be obtained.

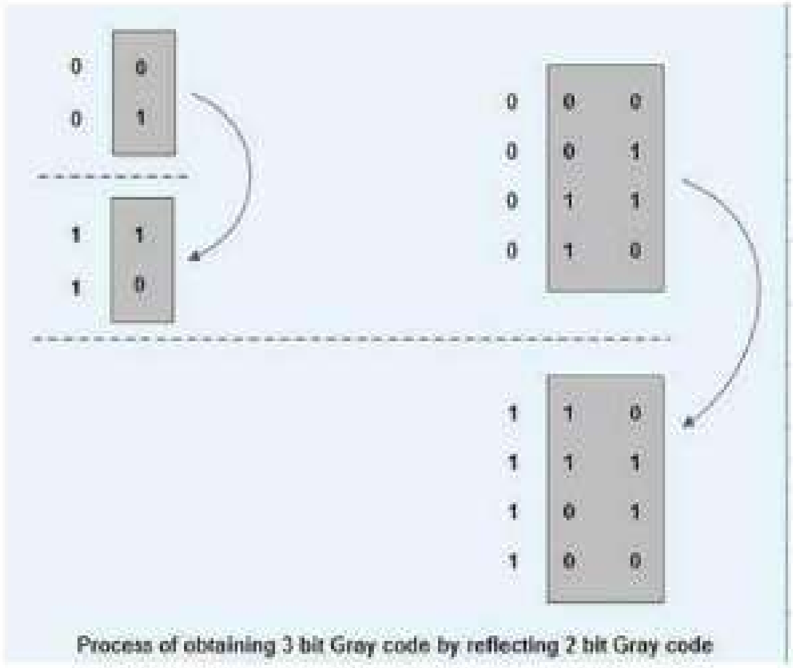

0 0
0 1
1 1
1 0
0 0 0
0 0 1
0 1 1
0 1 0
1 1 0
1 1 1
1 0 1
1 0 0
Process of obtaining 3 bit Gray code by reflecting 2 bit Gray code

Figure 1.2 Three-bit Gray code

Let us consider a few examples. The four-bit Gray code for decimal number 39 is 00101101.

Similarly, Gray code for (923.1)10 and (327) is

$$\overbrace{0010}^{3}\ \overbrace{1101}^{9}$$

(923.1)10 = (1101 0011 0010.0001) Gray code

(327)10 = (100011 0100) Gray code

Excess-3 code

Excess-3, also called XS3, is a non-weighted code used to express decimal number-s. It overcomes the shortcomings encountered while adding two decimal digits whose sum exceeds 9 as in case of 8421 BCD Code.

The Excess-3 code is formed by adding _3' to each digit and then replacing each digit by its four-bit binary equivalent.

The key feature of the Excess-3 code is .that it is self-complementing. In other words, the l's complement of an Excess-3 number is the Excess-3 code for the 9's complement of the corresponding decimal number. For example, the Excess-3 code for decimal 6 is (0110+0011)=1001. The l's complement of 1001 is 0110, which is the Excess-3 code for decimal

3 (0011+0011), and 3 is the 9's complement of 6. This property of Excess-3 code makes it usefulin some arithmetic operations.

Binary-coded decimal (BCD)

For many applications, we need the exact binary equivalent of the decimal system, which means we need an encoding for individual decimal digits. This is precisely the case in many business applications that deal with money—we cannot afford the rounding errors that occur when we convert real numbers to floating point during financial transactions!

Binary-coded decimal which is commonly referred to by its abbreviation **BCD** is very common in electronics, particularly those that display numerical data, such as alarm clocks and calculators.

Each decimal digit is individually converted to its binary equivalent. For example, to encode146, the decimal digits are replaced by 0001, 0100, and 0110, respectively. Because most computers use bytes as the smallest unit of access, most values are stored in 8 bits,not 4. That gives us two choices for storing 4-bit BCD digits.

Digit	**BCD**
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110

7	0111
8	1000
9	1001
Zones	
1111	Unsigned
1100	Positive
1101	Negative

Table 1.5 Binary-Coded Decimal

American Standard Code for Information Interchange (ASCII)

Digital computers deal with lot of alphanumeric information through people, input, and output devices. So, there is a need for a standard binary code for alphanumeric data. ASCII is a standard, which uses seven bits to code 128 characters in binary. The letter A, for example, is represented in ASCII as 1000001. The ASCII code contains the following characters:

- 94 printing characters and 34 nonprinting characters.
- The printing characters consist of the 26 uppercase letters A through Z, the 26 lowercase letters, the 10 numerals 0 through 9, and 32 special printable characters such as %, * , and
  $.
- The nonprinting characters or control characters are used for routing data and arranging the printed text into a prescribed format. Following are three types of control characters:

1. Format effectors: backspace (BS), horizontal tabulation (HT), and carriage return (CR)which control the printing layout.
2. Information separators: record separator (RS) and file separator (FS) which separate thedata into paragraphs and pages.
3. Communication control characters: STX (start of text) and ETX (end of text).

which areused to frame a text message when transmitted through a communication medium.

ASCII characters most often are stored one per byte, though it is a 7-bit code. Most computers manipulate an 8-bit quantity as a single unit called a byte. Therefore, ASCII characters most often are stored one per byte. The extra bit is sometimes used for other purposes, depending on the application. For example, some printers recognize 8-bit ASCII characters with the most significant bit set to 0. Additional 128 8-bit characters with the most significant bit set to 1 are used for other symbols, such as the Greek alphabet or italic type font. When used in data communication, the eighth bit may be employed to indicate the parity of the binary-coded character. It is possible to convert decimal digits in ASCII to BCD by removing the three high-order bits, 011.

Binary codes can be prepared for any set of discrete elements such as the musical notes andpositions on the chessboard.

Check Your Progress- 1

1. What is data type and number system?

....

....

....

....

2. What is the importance of number system?

....

....

....

....

3 What is the rule of number system?

....

....

....

....