1 **Digital Systems**

What is Digital System?

- **Digital System**: is a system in which signals have finite number of discrete values (electric impulses, decimal digits, arithmetic operations, etc.)
- **Analog System**: is a system in which signals have infinite number of values (electric voltage that vary with time).
- **Synchronous**: Systems where signals may change only at discrete instants.
- **Asynchronous**: Systems where signals may change at any instant.

Why Are Digital Systems important?

- It is well suited for numerical and non-numerical information processing.
- Information processing can use a general-purpose system (computer).
- The finite number of values in a digital signal is represented by a vector of signals with just 2 values (**binary signals**).

```
<table>
<thead>
<tr>
<th>digit</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>vector</td>
<td>0000</td>
<td>0001</td>
<td>0010</td>
<td>0011</td>
<td>0100</td>
<td>0101</td>
<td>0110</td>
<td>0111</td>
<td>1000</td>
<td>1001</td>
</tr>
</tbody>
</table>
```
- Digital signals are quite insensitive to variations of component variable values.

- Numerical digital systems can be made more accurate by increasing the number of digits used in the representation.
- Complex digital systems are built as integrated circuits composed of a large number of very simple devices.
- It is possible to select among different implementations of systems that trade off speed and amount of hardware.

**When Are Digital Systems Used?**

- Digital representation and processing methods widely used
- Extraordinary progress in digital technology and use Indispensable in modern society
- New applications fueled by the development of computer technology
- Knowledge about the design and use of digital systems required in a large variety of human activities
Analog and Digital Signals

- The process of converting from analog to digital is call quantization or digitization.

Combinational and Sequential Systems

- Digital systems are divided into 2 classes:
  o **Combinational systems**: the output at time $t$ depends only on the input at $t$.
  o $z(t) = F(x(t))$
  o In this case we can say that the system has no memory b/c the output doesn’t depend on previous inputs.
  o **Sequential systems**: the output at time $t$ depends on the input at time $t$ and possibly on the input prior to $t$.
  o $z(t) = F(x(0,t))$
  o *where* $x(0,t)$ is the input sequence from time 0 to time $t$. 
Binary Numbers

- A decimal number such as 7392 can be represented as:
  \[ 7 \times 10^3 + 3 \times 10^2 + 9 \times 10^1 + 2 \times 10^0 \]

- A number with a decimal point is represented by a series of coefficients as follows:
  \[ a_5 \ a_4 \ a_3 \ a_2 \ a_1 \ a_0 \ a_{-1} \ a_{-2} \ a_{-3} \]

- The decimal equivalent of the binary 11010.11 is 26.75
  \[ 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = 26.75 \]

- A number expressed in base-\(r\) system has coefficients multiplied by powers of \(r\):
  \[ a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \ldots + a_2 \cdot r^2 + a_1 \cdot r + a_0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2} + \ldots + a_{-m} \cdot r^{-m} \]
  where \(r = 2, 3, 4, \ldots, 8, 9, 10, \ldots, 16\ldots\)

<table>
<thead>
<tr>
<th>System</th>
<th>Radix</th>
<th>Allowable Digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>2</td>
<td>0,1</td>
</tr>
<tr>
<td>Octal</td>
<td>8</td>
<td>0,1,2,3,4,5,6,7</td>
</tr>
<tr>
<td>Decimal</td>
<td>10</td>
<td>0,1,2,3,4,5,6,7,8,9</td>
</tr>
<tr>
<td>Hexadecimal</td>
<td>16</td>
<td>0,1,2,3,4,5,6,7,8,9,A, B, C, D, E, F</td>
</tr>
</tbody>
</table>

- \((4021.2)_5 = \) 4 x 5^3 + 0 x 5^2 + 2 x 5^1 + 1 x 5^0 + 2 x 5^{-1} = (511.4)_{10}
  4 x 125 + 0 + 10 + 1 + 2 x (1/5)
  500 + 11 + .4

- \((B65F)_{16} = \) 11 x 16^3 + 6 x 16^2 + 5 x 16^1 + 15 x 16^0 = (46687)_{10}
  11 x 4096 + 6 x 256 + 5 x 16 + 15
  45056 + 1536 + 80 + 15

Augend: \[ 101101 \] minuend: \[ 101101 \] multiplicand: \[ 1011 \]
Addend: \[ + 100111 \] subtrahend: \[ -100111 \] multiplier: \[ \times 101 \]

\[ \underline{1010100} \quad \underline{000110} \quad \underline{110111} \]

Number Base Conversions

- A binary number can be converted to decimal by forming the sum of powers of 2 of those coefficients whose value is 1.
  \[ (1010.011)_2 = 2^3 + 2^1 + 2^{-2} + 2^{-3} = (10.375)_{10} \]

- Similarly, a number expressed in base \(r\) can be converted to its decimal equivalent by multiplying each coefficient with the corresponding power of \(r\) and adding.
  \[ (630.4)_8 = 6 \times 8^2 + 3 \times 8^1 + 0 \times 8^0 + 4 \times 8^{-1} = (408.5)_{10} \]
Conversion from Decimal 41 to Binary:

<table>
<thead>
<tr>
<th>Integer quotient</th>
<th>Remainder</th>
<th>Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>41/2 = 20</td>
<td>+ 1/2</td>
<td>$a_0 = 1$</td>
</tr>
<tr>
<td>20/2 = 10</td>
<td>+ 0</td>
<td>$a_1 = 0$</td>
</tr>
<tr>
<td>10/2 = 5</td>
<td>+ 0</td>
<td>$a_2 = 0$</td>
</tr>
<tr>
<td>5/2 = 2</td>
<td>+ 1/2</td>
<td>$a_3 = 1$</td>
</tr>
<tr>
<td>2/2 = 1</td>
<td>+ 0</td>
<td>$a_4 = 0$</td>
</tr>
<tr>
<td>1/2 = 0</td>
<td>+ 1/2</td>
<td>$a_5 = 1$</td>
</tr>
</tbody>
</table>

The conversion from decimal integers to any base-$r$ system is similar to the example, except that division is done by $r$ instead of 2.

Conversion from Decimal 153 to Octal:

\[
\begin{array}{c|c}
153 & 1 \\
19  & 3 \\
2   & 3 \\
0   & 2
\end{array}
\]

= (231)\textsubscript{8}

Conversion from Decimal fraction (0.6875)\textsubscript{10} to Binary:

<table>
<thead>
<tr>
<th>Integer</th>
<th>Fraction</th>
<th>Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6875 x 2 = 1</td>
<td>+ 0.3750</td>
<td>$a_{.1} = 1$</td>
</tr>
<tr>
<td>0.3750 x 2 = 0</td>
<td>+ 0.7500</td>
<td>$a_{.2} = 0$</td>
</tr>
<tr>
<td>0.7500 x 2 = 1</td>
<td>+ 0.5000</td>
<td>$a_{.3} = 1$</td>
</tr>
<tr>
<td>0.5000 x 2 = 1</td>
<td>+ 0.0000</td>
<td>$a_{.4} = 1$</td>
</tr>
</tbody>
</table>

The conversion from decimal fraction to any base-$r$ system is similar to the example. Multiplication is by $r$ instead of 2, and the coefficients found from the integers may range in value from 0 to $r-1$ instead of 0 and 1.

Conversion from Decimal fraction (0.513)\textsubscript{10} to Octal:

\[
\begin{align*}
0.513 & \times 8 = 4.104 \\
0.104 & \times 8 = 0.832 \\
0.832 & \times 8 = 6.656 \\
0.656 & \times 8 = 5.248 \\
0.248 & \times 8 = 1.984 \\
0.984 & \times 8 = 7.872
\end{align*}
\]

\[(0.513)_{10} = (0.406517\ldots)_{8}\]

The conversion of decimal numbers with both integers and fraction parts is done by converting the integer and fraction separately and then combining the two answers.
**Octal and Hexadecimal Numbers**

- The conversion from and to binary, octal and hexadecimal plays an important part in digital computers. Since $2^3 = 8$ and $2^4 = 16$, each octal digit corresponds to three binary digits and each hexadecimal digit corresponds to four binary digits.

- Conversion from binary to Octal:

  \[(10 110 001 101 011. 111 100 000 110) \_2 = (26153.7406) \_8\]

- Conversion from binary to Hexadecimal:

  \[(10 1100 0110 1011. 1111 0000 0110) \_2 = (2C6B.F06) \_16\]

- Conversion from Octal to binary:

  \[(673.124) \_8 = (110 111 011. 001 010 100) \_2\]

- Conversion from Hexadecimal to binary:

  \[(306.D) \_16 = (0011 0000 0110. 1101) \_2\]

- Conversion from Hexadecimal to Decimal:

  \[(37B) \_16 = 3 \times 16^2 + 7 \times 16^1 + 11 \times 16^0 = 3 \times 256 + 7 \times 16 + 11 \times 1 = 768 + 112 + 11 = (891) \_10\]

**Complements**

- Are used to simplify the subtraction operation and for logical manipulation.

**Diminished Radix Complement**

- Given a number $N$ in base $r$ having $n$ digits, the $(r - 1)$’s complement of $N$ is defined as $(r - 1) - N$. For decimal numbers, $r = 10$ and $r - 1 = 9$, so the ninth 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.

- If $n = 4 \Rightarrow 10^4 = 10,000$ and $10^4 - 1 = 9999$.

- The 9’s complement of 546700 is 999999 – 546700 = 453299

- The 9’s complement of 012398 is 999999 – 012398 = 987601

- For binary numbers, $r = 2$ and $r - 1 = 1$, so the 1’s complement of $N$ is $2^n - 1 - N$. $2^n$ is represented by a binary number that consists of a 1 followed by $n$ 0’s.
- If \( n = 4 \Rightarrow 2^4 = (10000)_2 \) and \( 2^4 - 1 = (1111)_2 \).

- The 1's complement of a binary number is formed by changing 1's to 0's and 0's to 1's.

The 1's complement of 1011000 is 0100111
The 1's complement of 0101101 is 1010010

**Radix Complement**

- The \( r \)'s complement of \( n \)-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, the \( r \)'s complement is obtained by adding 1 to the \( (r - 1) \)'s complement since \( r^n - N = [(r^n - 1) - N] + 1 \).

- The 10's complement of decimal \( 2389 = (10^4 - 1) - 2389 + 1 = 7611 \).
- The 2's complement of binary \( 101100 = (2^6 - 1) - 101100 + 1 = 010100 \).
- The 10's complement of decimal \( 012389 = (10^6 - 1) - 012389 + 1 = 987602 \).
- The 10's complement of decimal \( 246700 = (10^6 - 1) - 246700 + 1 = 753300 \).

- The 2's complement can be formed by leaving the least significant 0's and the first 1 unchanged, and the replacing 1's with 0's and 0's with 1's in the other four most-significant digits.

The 2's complement of binary \( 1101100 \) is 0010100.

- The 2’s complement of the following number is obtained by leaving the least significant 1 unchanged, and complementing all other digits.

The 2’s complement of binary \( 0110111 \) is 1001001.

**Summary**

- The radix complement and diminished radix complement are defined as:
  \[
  (N)_r = \text{an } n \text{-digit number } N \text{ in base } r \\
  [N]_r = \text{the } r \text{'s complement of } (N)_r \\
  [N]_{r-1} = \text{the } (r-1) \text{'s complement of } (N)_r \\
  [N]_r = r^n - (N)_r \quad \text{(Eq.1)} \\
  [N]_{r-1} = [N]_r - 1 \quad \text{(Eq.2)}
  \]

From Eq.1 and Eq.2, we can also derive the following equations:

\[
[N]_r = [N]_{r-1} + 1 \quad \text{(Eq.3)} \\
[N]_{r-1} = (r^n - 1) - (N)_r \quad \text{(Eq.4)}
\]
In the decimal system, \( r=10 \), we have 10’s complement and 9’s complement. In the octal system, \( r=8 \), we have 8’s complement and 7’s complement. In the binary system, \( r=2 \), we have 2’s complement and 1’s complement.

<table>
<thead>
<tr>
<th>System</th>
<th>Radix Complement</th>
<th>Diminished Radix Complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>10’s complement</td>
<td>9's complement</td>
</tr>
<tr>
<td>Octal</td>
<td>8’s complement</td>
<td>7's complement</td>
</tr>
<tr>
<td>Binary</td>
<td>2’s complement</td>
<td>1’s complement</td>
</tr>
</tbody>
</table>

To find the radix complement representation of a number, it is more convenient to first derive the diminished radix complement. The radix complement is then obtained by adding 1 to the diminished radix complement.

The 9’s complement of a decimal number is obtained by subtracting each digit from 9. The 7’s complement of an octal number is obtained by subtracting each digit from 7. The 1’s complement of a binary integer is obtained by subtracting each digit from 1.

Examples:

- Find the 10's complement and the 9's complement of \((546700)_{10}\)
  
  \[
  \begin{align*}
  (453299)_{10} & \quad 9's \ complement \\
  (453300)_{10} & \quad 10's \ complement \quad (\text{add 1 to the 9's complement})
  \end{align*}
  \]

- Find the 8’s complement and the 7’s complement of \((526071)_{8}\)
  
  \[
  \begin{align*}
  (251706)_{8} & \quad 7's \ complement \\
  (251707)_{8} & \quad 8's \ complement \quad (\text{add 1 to the 7's complement})
  \end{align*}
  \]

- Find the 2’s complement and the 1’s complement of \((00011010)_{2}\)
  
  \[
  \begin{align*}
  (11100110)_{2} & \quad 1's \ complement \\
  (11100101)_{2} & \quad 2's \ complement \quad (\text{add 1 to the 1's complement})
  \end{align*}
  \]

**Subtraction with Complements**

Example (1):

Using 10’s complement, subtract \(72532 - 3250\)

\[
M - N
\]

\[
\begin{align*}
M &= 72532 \\
10's \ complement \ of \ N &= +96750 \quad (99999 - 03250) + 1 \\
\text{Sum} &= 169282 \\
\text{Discard end carry} \ 10^5 &= -100000 \\
\text{Answer} &= 69282
\end{align*}
\]
Example (2):
Using 10’s complement, subtract $3250 - 72532$

\[ M = N \]

\[ M = 03250 \]

10’s complement of $N = 27468 \quad (99999 - 72532) + 1$

Sum = 30718

No end carry.

Answer - (10’s complement of 30718) = -69282

Example (3):
Using 2’s complement, subtract $1010100 - 1000011$

\[ X - Y \]

\[ X = 1010100 \]

2’s complement of $Y = 0111101$

Sum = 10010001

Discard end carry $2^7$ = -10000000

Answer: $X - Y = 0010001$

Example (4):
Using 2’s complement, subtract $1000011 - 1010100$

\[ Y - X \]

\[ Y = 1000011 \]

2’s complement of $X = 0101100$

Sum = 1101111

No end carry.

Answer: $Y - X - (2’s \text{ complement of } 1101111) = -0010001$

Example (5): Using 1’s complement, subtract $X - Y = 1010100 - 1000011$

\[ X = 1010100 \]

1’s complement of $Y = 0111100$ (+1 End-around carry)

Sum = 10010000

Answer: $X - Y = 0010001$

Example (6): Using 1’s complement, subtract $Y - X = 1000011 - 1010100$

\[ Y = 1000011 \]

1’s complement of $X = 0101011$

Sum = 1101110

No end carry.

Answer: $Y - X - (1’s \text{ complement of } 1101110) = -0010001$
Signed Binary Numbers

- It is customary to represent the sign with a bit placed in the leftmost position of the number and to make it 0 for positive and 1 for negative.

- Consider the number 9 represented in binary with 8 bits. +9 is represented with sign bit 0 in the leftmost position followed by the binary equivalent of 9 to give 00001001.

<table>
<thead>
<tr>
<th>Signed magnitude</th>
<th>Signed-1’s complement</th>
<th>Signed-2’s complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001001</td>
<td>11110110</td>
<td>11110111</td>
</tr>
</tbody>
</table>

Arithmetic Addition

- The addition of 2 numbers in the signed-magnitude system follows the rules of ordinary arithmetic.
- If the signs are the same, we add the two magnitudes and give the sum the common sign.
- If the signs are different, we subtract the smaller magnitudes from the larger and give the result the sign of the larger magnitude.

| +6     | 00000110 | -6    | 11111010 |
| +13    | 00001101 | +13   | 00001101 |
| 00010011 |         | 00001111 |

| +6     | 00000110 | -6    | 11111010 |
| -13    | 11110011 | -13   | 11110011 |
| 11111001 |         | 11101101 |

- Negative numbers must be in 2’s complement and that the sum obtained after the addition if negative is in 2’s-complement form.

Arithmetic Subtraction

- Take the 2’s complement of the subtrahend (including the sign bit) and add it to the minuend (including the sign bit). A carry out of the sign-bit position is discarded.

  \[(\pm A) - (+B) = (\pm A) + (-B)\]
  \[(\pm A) - (-B) = (\pm A) + (+B)\]

Binary Codes

- Binary codes play an important role in digital computers. A bit is a binary digit. It is equal to 0 or 1.
- Although the minimum number of bits required to code \(2^n\) quantities is \(n\), there is no maximum number of bits that may be used for a binary code.

Decimal Codes

- BCD (binary-code decimal) is a straight assignment of the binary equivalent. Check Table 1-2 page 18.
Error Detection Code
- Binary information can be transmitted from one location to another. External noise may change some of the bits from 0 to 1 and vice versa. To achieve error-detection we use a parity bit.
- A parity bit is an extra bit included with a message to the total number of 1’s transmitted either odd or even. See Table 1-3 Page 20.
- Two methods are implemented:
  o Even Parity: the P bit is chosen so that the total number of 1’s in the five bits is even.
  o Odd Parity: the P bit is chosen so that the total number of 1’s in the five bits is odd.

Gray Code
- Gray code is used to represent the digital data when it is converted from analog data. See Table 1 – 4 Page 21.
- The advantage of the Gray code over binary numbers is that only one bit in the code group changes when going from one number to the next.
- In Gray to go from 7 to 8: 0100 → 1100
  In Binary to from: 7 to 8: 0111 → 1000

ASCII Character Code
- The standard binary code for representation of alphanumeric characters is ASCII (American Standard Code for Information Interchange). It uses 7 bits to code 128 characters. See Table 1-5 Page 23.

Binary Storage & Registers
Register
- A register is a group of binary cells. Each cell stores one bit of information. The state of a register is an n—tuple of 1’s and 0’s, with each bit designating the state of one cell in the register.
- The content of a register is a function of the interpretation given to the information stored in it. See page 25.

Binary Logic
- Deals with variables that take on two discrete values and with operations that assume logical meaning.
- The 2 values may be called by different names (e.g. true/false, yes/no, 0/1)
- It is suited for the analysis and design of digital systems.
Definition of Binary Logic
- Consists of binary variables and logical operations.
  - **Variables**: A, B, C, x, y, z, etc., with each variable having two values 1 and 0
  - **Logical Operations**:
    - **AND**: is represented by a dot or an absence of an operator.
      - EX: \(x \cdot y = z\) or \(xy = z\) or is read as “\(x\) AND \(y\) is equal to \(z\).”
      - It means \(z = 1\) if \(x = 1\) and \(y = 1\); otherwise, \(z = 0\).
    - **OR**: is represented by + sign.
      - EX: \(x + y = z\) or is read as “\(x\) OR \(y\) is equal to \(z\).”
      - It means \(z = 1\) if \(x = 1\) or \(y = 1\) or if \(x = 1\) and \(y = 1\). If both \(x = 0\) and \(y = 0\), then \(z = 0\).
    - **NOT**: is represented by ‘ or ‘.
      - EX: \(x' = z\) (or \(\bar{0} = z\)) is read, “not \(x\) is equal to \(z\),” meaning that \(z\) is what \(x\) is not. In other words, if \(x = 1\), then \(z = 0\); but if \(x = 0\), then \(z = 1\).
    - For each combination of values \(x\) and \(y\), there is a value \(z\) specified by the definition of the logical operation. The definition is listed in compact form using **truth tables**.

**Truth Table of Logical Operations**

<table>
<thead>
<tr>
<th></th>
<th>AND</th>
<th></th>
<th>OR</th>
<th></th>
<th>NOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)</td>
<td>(y)</td>
<td>(x \cdot y)</td>
<td>(x)</td>
<td>(y)</td>
<td>(x + y)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Logic Gates**
- Electronic digital circuits are called logic circuits because, with the proper input, they establish logical manipulation paths. See Fig. 1 – 6 Page 31.

Two-input AND gate

Two-input OR gate

Not gate or inverter