Learning Objectives

- To study Binary, Octal, Hexadecimal, Decimal number systems.
- Conversion of Binary to Octal, Binary to decimal, Binary to Hexadecimal and Conversion.
- Binary Addition, Subtraction, Multiplication, and Division.
- To study the conversion of 1’s, 2’s Complements.
- To study ASCII and EBCDIC codes.
- To study Boolean Algebra and its postulates.

1.0 Introduction

Binary, hexadecimal, and octal refer to different number systems. The one that we typically use is called decimal. These number systems refer to the number of symbols used to represent numbers. In the decimal system, we use ten different symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. With these ten symbols, we can represent any quantity. For example, if we see a 2, then we know that there is two of something. For example, this sentence has 2 periods on the end.

When we run out of symbols, we go to the next digit placement. To represent one higher than 9, we use 10 meaning one unit of ten and zero units of one. This may seem elementary, but it is crucial to understand our default number system if you want to understand other number systems.
For example, when we consider a binary system which only uses two symbols, 0 and 1, when we run out of symbols, we need to go to the next digit placement. So, we would count in binary 0, 1, 10, 11, 100, 101, and so on.

1.1 BINARY NUMBER SYSTEM

Number systems are used to describe the quantity of something or represent certain information. Because of this, I can say that the word “calculator” contains ten letters. Our number system, the decimal system, uses ten symbols. Therefore, decimal is said to be Base Ten. By describing systems with bases, we can gain an understanding of how that particular system works.

When we count in Base Ten, we count starting with zero and going up to nine in order.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, …

Once we reach the last symbol, we create a new placement in front of the first and count that up.

8, 9, 10, 11, 12, … , 19, 20, …

This continues when we run out of symbols for that placement. So, after 99, we go to 100.

The placement of a symbol indicates how much it is worth. Each additional placement is an additional power of 10. Consider the number of 2853. We know this number is quite large, for example, if it pertains to the number of apples in a basket. That’s a lot of apples. How do we know it is large? We look at the number of digits.

Each additional placement is an additional power of 10, as stated above. Consider this chart.

<table>
<thead>
<tr>
<th>10^3</th>
<th>10^2</th>
<th>10^1</th>
<th>10^0</th>
</tr>
</thead>
<tbody>
<tr>
<td>digit</td>
<td>digit</td>
<td>digit</td>
<td>digit</td>
</tr>
<tr>
<td>*1000</td>
<td>*100</td>
<td>*10</td>
<td>*1</td>
</tr>
</tbody>
</table>

1.1.1 Binary Number System

Binary is another way of saying Base Two. So, in a binary number system, there are only two symbols used to represent numbers: 0 and 1. When
we count up from zero in binary, we run out of symbols much more frequently.

\[ 0, 1, \ldots \]

From here, there are no more symbols. We do not go to 2 because in binary, a 2 doesn’t exist. Instead, we use 10. In a binary system, 10 is equal to 2 in decimal.

We can count further.

<table>
<thead>
<tr>
<th>Binary</th>
<th>0</th>
<th>1</th>
<th>10</th>
<th>11</th>
<th>100</th>
<th>101</th>
<th>110</th>
<th>111</th>
<th>1000</th>
<th>1001</th>
<th>1010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

Just like in decimal, we know that the more digits there are, the larger the number. However, in binary, we use powers of two. In the binary number 1001101, we can create a chart to find out what this really means.

\[
\begin{array}{c|c|c|c|c|c|c|c|c}
2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 \\
\end{array}
\]

\[ 64 + 0 + 0 + 8 + 4 + 0 + 1 \]

\[ 87 \]

Since this is base two, however, the numbers don’t get quite as large as it does in decimal. Even still, a binary number with 10 digits would be larger than 1000 in decimal.

The binary system is useful in computer science and electrical engineering. Transistors operate from the binary system, and transistors are found in practically all electronic devices. A 0 means no current, and a 1 means to allow current. With various transistors turning on and off, signals and electricity is sent to do various things such as making a call or putting these letters on the screen.

Computers and electronics work with bytes or eight digit binary numbers. Each byte has encoded information that a computer is able to understand. Many bytes are stringed together to form digital data that can be stored for use later.

1.1.2 Octal Number System

Octal is another number system with less symbols to use than our conventional number system. Octal is fancy for Base Eight meaning eight symbols are used to represent all the quantities. They are 0, 1, 2, 3, 4, 5, 6, and 7. When
we count up one from the 7, we need a new placement to represent what we call 8 since an 8 doesn’t exist in Octal. So, after 7 is 10.

<table>
<thead>
<tr>
<th>Octal</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>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
</tbody>
</table>

Just like how we used powers of ten in decimal and powers of two in binary, to determine the value of a number we will use powers of 8 since this is Base Eight. Consider the number 3623 in base eight.

\[
\begin{array}{cccc}
8^3 & 8^2 & 8^1 & 8^0 \\
3 & 6 & 2 & 3 \\
1536 + 384 + 16 + 3 \\
1939
\end{array}
\]

Each additional placement to the left has more value than it did in binary. The third digit from the right in binary only represented \(2^{3-1}\), which is 4. In octal, that is \(8^{3-1}\) which is 64.

### 1.1.3 Hexadecimal Number System

The hexadecimal system is Base Sixteen. As its base implies, this number system uses sixteen symbols to represent numbers. Unlike binary and octal, hexadecimal has six additional symbols that it uses beyond the conventional ones found in decimal. But what comes after 9? 10 is not a single digit but two… Fortunately, the convention is that once additional symbols are needed beyond the normal ten, letters are to be used. So, in hexadecimal, the total list of symbols to use is 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. In a digital display, the numbers B and D are lowercase.

When counting in hexadecimal, you count 0, 1, 2, and so on. However, when you reach 9, you go directly to A. Then, you count B, C, D, E, and F. But what is next? We are out of symbols! When we run out of symbols, we create a new digit placement and move on. So after F is 10. You count further until you reach 19. After 19, the next number is 1A. This goes on forever.
Digits are explained as powers of 16. Consider the hexadecimal number 2DB7.

<table>
<thead>
<tr>
<th>Hexadecimal</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
</tbody>
</table>

\[ 2 \times 16^3 + D \times 16^2 + B \times 16^1 + 7 \times 16^0 = 8192 + 3328 + 176 + 7 = 11703 \]

As you can see, placements in hexadecimal are worth a whole lot more than in any of the other three number systems.

### 1.2 Conversion of Number System

It is important to know that 364 in octal is not equal to the normal 364. This is just like how a 10 in binary is certainly not 10 in decimal. 10 in binary (this will be written as 10\(_2\) from now on) is equal to 2. 10\(_8\) is equal to 8. How on earth do we know this? What is 20C.38F\(_{16}\) and how do we find out?

Here is why it is important to understand how the number systems work. By using our powers of the base number, it becomes possible to turn any number to decimal and from decimal to any number.

**Binary to Decimal Conversion**

- Step 1: Check if your number is odd or even.
- Step 2: If it’s even, write 0 (proceeding backwards, adding binary digits to the left of the result).
- Step 3: Otherwise, if it’s odd, write 1 (in the same way).
- Step 4: Divide your number by 2 (dropping any fraction) and go back to step 1. Repeat until your original number is 0.
An example:

Convert 68 to binary:
- 68 is even, so we write 0.
- Dividing 68 by 2, we get 34.
- 34 is also even, so we write 0 (result so far - 00)
- Dividing 34 by 2, we get 17.
- 17 is odd, so we write 1 (result so far - 100 - remember to add it on the left)
- Dividing 17 by 2, we get 8.5, or just 8.
- 8 is even, so we write 0 (result so far - 0100)
- Dividing 8 by 2, we get 4.
- 4 is even, so we write 0 (result so far - 00100)
- Dividing 4 by 2, we get 2.
- 2 is even, so we write 0 (result so far - 000100)
- Dividing 2 by 2, we get 1.
- 1 is odd, so we write 1 (result so far - 1000100)
- Dividing by 2, we get 0.5 or just 0, so we’re done.
- Final result: 1000100

From Binary to Decimal
- Write the values in a table as shown before. (or do so mentally)
- Add the value in the column header to your number, if the digit is turned on (1).
- Skip it if the value in the column header is turned off (0).
- Move on to the next digit until you’ve done them all.

An example:

Convert 101100 to decimal:
- Highest digit value: 32. Current number: 32
- Skip the “16” digit, its value is 0. Current number: 32
· Add 8. Current number: 40
· Add 4. Current number: 44
· Skip the “2” and “1” digits, because their value is 0.
· Final answer: 44

From Decimal to Hexadecimal.

This is only one of the many ways!

· Convert your decimal number to binary
· Split up in nibbles of 4, starting at the end
· Look at the first table on this page and write the right number in place of the nibble

(you can add zeroes at the beginning if the number of bits is not divisible by 4, because, just as in decimal, these don’t matter)

Example

Convert 39 to hexadecimal:

· First, we convert to binary (see above). Result: 100111
· Next, we split it up into nibbles: 0010/0111 (Note: I added two zeroes to clarify the fact that these are nibbles)
· After that, we convert the nibbles separately.
· Final result: 27

From Hexadecimal to Decimal

Check the formula in the first paragraph and use it on the ciphers in your hexadecimal number. (this actually works for any conversion to decimal notation)

Example

Convert 1AB to Decimal

· Value of B = 16^0*11. This gives 11, obviously
· Value of A = 16^1*10. This gives 160. Our current result is 171.
· Value of 1 = 16^2*1. This gives 256.
· Final result: 427
From Decimal to Octal

- Convert to binary.
- Split up in parts of 3 digits, starting on the right.
- Convert each part to an octal value from 1 to 7

**Example:** Convert 25 to octal

First, we convert to binary. Result: 11001

Next, we split up: 011/001

Conversion to octal: 31

From Octal to Decimal

Again, apply the formula from above

Example: Convert 42 to Decimal

Value of 2 = \(8^0 \times 2 = 2\)

Value of 4 = \(8^1 \times 4 = 32\)

Result: 34

1.4 Binary Arithmetic

The arithmetic operations - addition, subtraction, multiplication and division, performed on the binary numbers is called *binary arithmetic*. In computer systems, the basic arithmetic operations performed on the binary numbers is -

- Binary Addition, and
- Binary Subtraction,

In the following subsections, we discuss the binary addition and the binary subtraction operations.

1.4.1 Binary - Addition

Binary addition involves adding of two or more binary numbers. The binary addition rules are used while performing the binary addition. Table 1 shows the binary addition rules.
Table 1. Binary Addition Rules for two inputs

Binary addition of three inputs, when all the inputs are 1, follows the rule shown in Table 2.

<table>
<thead>
<tr>
<th>Input 1</th>
<th>Input 2</th>
<th>Sum</th>
<th>Carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No carry</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>No carry</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>No carry</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1.1 Binary Addition Rules

Binary addition of three inputs, when all the inputs are 1, follows the rule shown in Table 2.

<table>
<thead>
<tr>
<th>Input 1</th>
<th>Input 2</th>
<th>Input 3</th>
<th>Sum</th>
<th>Carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2 Binary Addition of three inputs

Addition of the binary numbers involves the following steps-

1. Start addition by adding the bits in unit column (the rightmost column). Use the rules of binary addition.
2. The result of adding bits of a column is a sum with or without a carry.
3. Write the sum in the result of that column.
4. If carry is present, the carry is carried-over to the addition of the next left column.
5. Repeat steps 2-4 for each column, i.e., the tens column, hundreds column and so on.

Let us now understand binary addition with the help of some examples.

**Example 1.** Add 10 and 01. Verify the answer with the help of decimal addition.

When we add 0 and 1 in the unit column, sum is 1 and there is no carry. The sum 1 is written in the unit column of the result. In the tens column, we add
1 and 0 to get the sum 1. There is no carry. The sum 1 is written in the tens column of the result.

1.4.2 Binary Subtraction

Binary subtraction involves subtracting two binary numbers. The binary subtraction rules are used while performing the binary subtraction. The binary subtraction rules are shown in Table 3.

<table>
<thead>
<tr>
<th>Input 1</th>
<th>Input 2</th>
<th>Difference</th>
<th>Borrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No borrow</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>No borrow</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>No borrow</td>
</tr>
</tbody>
</table>

The steps for performing subtraction of the binary numbers are as follows-

1. Start subtraction by subtracting the bit in the lower row from the upper row, in the unit column.
2. Use the binary subtraction rules. If the bit in the upper row is less than the lower row, borrow 1 from the upper row of the next column (on the left side). The result of subtracting two bits is the difference.
3. Write the difference in the result of that column.
4. Repeat step 2-3 for each column, i.e., the tens column, hundreds column and so on.

Let us now understand binary subtraction with the help of some examples.

**Example 1.** Subtract 01 from 11. Verify the answer with the help of decimal subtraction. When we subtract 1 from 1 in the unit column, the difference
is 0. Write the difference in the unit column of the result. In the tens column, subtract 0 from 1 to get the difference 1. Write the difference in the tens column of the result.

1.4.3 Binary Multiplication

It is actually much simpler than decimal multiplication. In the case of decimal multiplication, we need to remember \(3 \times 9 = 27, 7 \times 8 = 56\), and so on. In binary multiplication, we only need to remember the following,

\[
\begin{align*}
0 \times 0 &= 0 \\
0 \times 1 &= 0 \\
1 \times 0 &= 0 \\
1 \times 1 &= 1
\end{align*}
\]

Note that since binary operates in base 2, the multiplication rules we need to remember are those that involve 0 and 1 only. As an example of binary multiplication we have 101 times 11,

\[
\begin{array}{c}
101 \\
\times 11
\end{array}
\]

First we multiply 101 by 1, which produces 101. Then we put a 0 as a placeholder as we would in decimal multiplication, and multiply 101 by 1, which produces 101.

\[
\begin{array}{c}
101 \\
\times 11
\end{array}
\]

1010 ←— the 0 here is the placeholder

The next step, as with decimal multiplication, is to add. The results from
our previous step indicates that we must add 101 and 1010, the sum of which is 
1111.

\[
\begin{array}{c}
101 \\
x11 \\
101 \\
1010 \\
1111 \\
\end{array}
\]

1.4.4 Binary Division

It is almost as easy, and involves our knowledge of binary multiplication. Take for example the division of 1011 into 11.

\[
\begin{array}{c}
11 \text{ R=10} \\
11 \div 1011 \\
-11 \\
101 \\
-11 \\
10 \leftarrow \text{remainder, R} \\
\end{array}
\]

To check our answer, we first multiply our divisor 11 by our quotient 11. Then we add its' product to the remainder 10, and compare it to our dividend of 1011.

\[
\begin{array}{c}
11 \\
x11 \\
11 \\
11 \\
1001 \leftarrow \text{product of 11 and 11} \\
1001 \\
+ 10 \\
1011 \leftarrow \text{sum of product and remainder} \\
\end{array}
\]

The sum is equal to our initial dividend, therefore our solution is correct.
1.5 Complement of Binary Numbers

Complements are used in computer for the simplification of the subtraction operation. We now see, how to find the complement of a binary number. There are two types of complements for the binary number system – 1’s complement and 2’s complement.

1’s complement of Binary number is computed by changing the bits 1 to 0 and the bits 0 to 1. For example,

- 1’s complement of 110 is 001
- 1’s complement of 1011 is 0100
- 1’s complement of 1101111 is 0010000

2’s complement of Binary number is computed by adding 1 to the 1’s complement of the binary number. For example,

- 2’s complement of 110 is 001 + 1 = 010
- 2’s complement of 1011 is 0100 + 1 = 0101
- 2’s complement of 1101111 is 0010000 + 1 = 0010001

1.6 Subtraction of Binary Numbers in 2’S Complement

Addition of signed binary numbers – The addition of any two signed binary numbers is performed as follows -

- Represent the positive number in binary form. (For e.g., +5 is 0000 0101 and +10 is 0000 1010)
- Represent the negative number in 2’s complement form. (For e.g., -5 is 1111 1011 and -10 is 1111 0110)
- Add the bits of the two signed binary numbers.
- Ignore any carry out from the sign bit position.

Please note that the negative output is automatically in the 2’s complement form. We get the decimal equivalent of the negative output number, by finding its 2’s complement, and attaching a negative sign to the obtained result. Let’s understand the addition of two signed binary numbers with the help of some examples.

Example 1. Add +5 and +10.

We represent +5 in binary form, i.e., 0000 0101. We represent +10 in
binary form, i.e., 0000 1010. Add the two numbers. The result is 0000 1111 i.e. +15.

1.6.2 Substraction of Signed Binary Numbers

The subtraction of signed binary numbers is changed to the addition of two signed numbers. For this, the sign of the second number is changed before performing the addition operation.

\[
\begin{align*}
(-A) - (+B) &= (-A) + (-B) \quad (+B \text{ in subtraction is changed to } -B \text{ in addition}) \\
(+A) - (+B) &= (+A) + (-B) \quad (+B \text{ in subtraction is changed to } -B \text{ in addition}) \\
(-A) - (-B) &= (-A) + (+B) \quad (-B \text{ in subtraction is changed to } +B \text{ in addition}) \\
(+A) - (-B) &= (+A) + (+B) \quad (-B \text{ in subtraction is changed to } +B \text{ in addition})
\end{align*}
\]

We see that the subtraction of signed binary numbers is performed using the addition operation.

The hardware logic for the fixed point number representation is simple, when we use 2’s complement for addition and subtraction of the signed binary numbers.

1.7 Uses of Alpha Numeric Codes (ASCII & EBCDIC)

The alphabetic data, numeric data, alphanumeric data, symbols, sound data and video data, all are represented as combination of bits in the computer. The bits are grouped in a fixed size, such as 8 bits, 6 bits or 4 bits. A code is made by combining bits of definite size. Binary Coding schemes represent the data such as alphabets, digits 0-9, and symbols in a standard code. A combination of bits represents a unique symbol in the data. The standard code enables any programmer to use the same combination of bits to represent a symbol in the data.
The binary coding schemes that are most commonly used are -

- Extended Binary Coded Decimal Interchange Code (EBCDIC),
- American Standard Code for Information Interchange (ASCII), and

### 1.7.1 ASCII (American Standard Code for Information Interchange)

- The American Standard Code for Information Interchange (ASCII) is widely used in computers of all types.
- ASCII codes are of two types – ASCII-7 and ASCII-8.

  - **ASCII-7** is a 7-bit standard ASCII code. In ASCII-7, the first 3 bits are the zone bits and the next 4 bits are for the digits. ASCII-7 allows $2^7 = 128$ combinations. 128 unique symbols are represented using ASCII-7. ASCII-7 has been modified by IBM to ASCII-8.

  - **ASCII-8** is an extended version of ASCII-7. ASCII-8 is an 8-bit code having 4 bits for zone and 4 bits for the digit. ASCII-8 allows $2^8 = 256$ combinations. ASCII-8 represents 256 unique symbols. ASCII is used widely to represent data in computers.

    - The ASCII-8 code represents 256 symbols.
    - Codes 0 to 31 represent control characters (non-printable), because they are used for actions like, Carriage return (CR), Bell (BEL) etc.
        - Codes 48 to 57 stand for numeric 0-9.
        - Codes 65 to 90 stand for uppercase letters A-Z.
        - Codes 97 to 122 stand for lowercase letters a-z.
        - Codes 128-255 are the extended ASCII codes.

### 1.7.2 EBCDIC (Extended Binary Coded Decimal Interchange Code)

- The Extended Binary Coded Decimal Interchange Code (EBCDIC) uses 8 bits (4 bits for zone, 4 bits for digit) to represent a symbol in the data.
- EBCDIC allows $2^8 = 256$ combinations of bits.
- 256 unique symbols are represented using EBCDIC code. It represents decimal numbers (0-9), lower case letters (a-z), uppercase letters (A-Z), Special characters, and Control characters (printable and non-printable e.g. for cursor movement, printer vertical spacing etc.).
- EBCDIC codes are used, mainly, in the mainframe computers.
1.8 Postulates of Boolean Algebra

Assume A, B, and C are logical states that can have the values 0 (false) and 1 (true).

“+” means OR, “·” means AND, and NOT[A] means NOT A.

POSTULATES

<table>
<thead>
<tr>
<th>Postulate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) A + 0 = A</td>
<td>A · 1 = A</td>
</tr>
<tr>
<td>(2) A + NOT[A] = 1</td>
<td>A · NOT[A] = 0</td>
</tr>
<tr>
<td>(3) A + B = B + A</td>
<td>A · B = B · A</td>
</tr>
<tr>
<td>(4) A + (B + C) = (A + B) + C</td>
<td>A · (B · C) = (A · B) · C</td>
</tr>
<tr>
<td>(5) A + (B · C) = (A + B) · (A + C)</td>
<td>A · (B + C) = (A · B) + (A · C)</td>
</tr>
</tbody>
</table>

Short Answer Type Questions

1. Mention the Number system used in Computers.
2. Mention the names of Number system.
3. Define 1’s Compliment of 1001, 1011.
4. Write 2’s Compliment of 1101, 1100.
5. Perform Binary addition of 1011, 0111.
6. Convert 99 into Binary and Octal numbers.
7. Convert 9A into Binary and Octal numbers.
8. Convert 48₁₀ into Hexadecimal number.

Long Answer Type Questions

1. Describe Binary, Octal, Decimal and Hexadecimal number system.
2. Explain inter conversion of Decimal to Binary, Hexadecimal to Binary number systems.
3. Convert Binary to Octal, Binary to Decimal number systems.
4. Explain briefly Binary Addition, Subraction, Multiplication and Division.
5. Explain postulates of Boolean Algebra.
On Job Training

1. Practice Conversion of Binary to Decimal and Decimal to Binary number system.

2. Practice Conversion of Binary to Octal and Hexadecimal system and vice versa.

3. Practice Postulates of Boolean Algebra
Learning Objectives

- To study Logic gates OR, AND, NOT operations with Truth table.
- To study working of NAND, NOR gates with truth table.
- To study working of Ex - OR gate with truth table.
- To study Demorgan Theorems
- To study the differences of Digital Logic families
- To study characteristics of Digital IC logic levels, Propagation delay, Noise margin, Fan-in, Fan-out and power dissipation.
- To study the comparisons of TTL, CMOS and ECL logic families.
- To study the Digital IC logic gates and their IC numbers.

2.0 Introduction

A logic gate is an elementary building block of a digital circuit. Most logic gates have two inputs and one output. At any given moment, every terminal is in one of the two binary conditions low (0) or high (1), represented by different voltage levels. The logic state of a terminal can, and generally does, change often, as the circuit processes data. In most logic gates, the low state is approximately zero volts (0 V), while the high state is approximately five volts positive (+5 V).
There are seven basic logic gates: AND, OR, XOR, NOT, NAND, NOR, and XNOR.

2.1 Explanation of AND, OR, NOT Logic Gates with Truth Table

2.1.1 AND Gate

The AND gate is so named because, if 0 is called “false” and 1 is called “true,” the gate acts in the same way as the logical “and” operator. The following illustration and table show the circuit symbol and logic combinations for an AND gate. (In the symbol, the input terminals are at left and the output terminal is at right.) The output is “true” when both inputs are “true.” Otherwise, the output is “false.”

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

2.1.2 OR Gate

The OR gate gets its name from the fact that it behaves after the fashion of the logical inclusive “or.” The output is “true” if either or both of the inputs are “true.” If both inputs are “false,” then the output is “false.”
2.1.3 NOT Gate

A logical inverter, sometimes called a NOT gate to differentiate it from other types of electronic inverter devices, has only one input. It reverses the logic state.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 2.4 OR Gate Truth table

![NOT Gate](image)

Fig. 2.5 NOT Gate

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Y</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig. 2.6 NOT Gate Truth table

2.2 Explanation of Universal Logic Gates: NAND, NOR with Truth Table

A universal gate is a gate which can implement any Boolean function without need to use any other gate type.

The NAND and NOR gates are universal gates.

In practice, this is advantageous since NAND and NOR gates are economical and easier to fabricate and are the basic gates used in all IC digital logic families.
In fact, an AND gate is typically implemented as a NAND gate followed by an inverter not the other way around!!

Likewise, an OR gate is typically implemented as a NOR gate followed by an inverter not the other way around

2.2.1 Universal NAND Gate, Working

To prove that any Boolean function can be implemented using only NAND gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

The figure shows two ways in which a NAND gate can be used as an inverter (NOT gate).

1. All NAND input pins connect to the input signal $A$ gives an output $A'$.

![Fig. 2.7 NAND Gate followed by Inverter with Single Input](image)

2. One NAND input pin is connected to the input signal $A$ while all other input pins are connected to logic 1. The output will be $A'$.

![Fig. 2.8 NAND Gate followed by Inverter with Two Inputs](image)

Implementing AND Using only NAND Gates

An AND gate can be replaced by NAND gates as shown in the figure (The AND is replaced by a NAND gate with its output complemented by a NAND gate inverter).

![Fig. 2.9 Two Inputs NAND Gates and Single Input NAND Gate is Equivalent to Two Inputs AND Gate](image)
Implementing OR Using only NAND Gates

An OR gate can be replaced by NAND gates as shown in the figure (The OR gate is replaced by a NAND gate with all its inputs complemented by NAND gate inverters).

![Diagram showing OR gate replaced by NAND gates](image)

Thus, the NAND gate is a universal gate since it can implement the AND, OR and NOT functions.

NAND Gate is a Universal Gate:

To prove that any Boolean function can be implemented using only NOR gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

2.2.2 UNIVERSAL NOR GATE, WORKING

Implementing an Inverter Using only NOR Gate

The figure shows two ways in which a NOR gate can be used as an inverter (NOT gate).

1. All NOR input pins connect to the input signal $A$ gives an output $A'$.

![Diagram showing NOR gate as inverter](image)

2. One NOR input pin is connected to the input signal $A$ while all other input pins are connected to logic 0. The output will be $A'$.
Implementing OR Using only NOR Gates

An OR gate can be replaced by NOR gates as shown in the figure (The OR is replaced by a NOR gate with its output complemented by a NOR gate inverter).

Implementing AND Using only NOR Gates

An AND gate can be replaced by NOR gates as shown in the figure (The AND gate is replaced by a NOR gate with all its inputs complemented by NOR gate inverters).

Thus, the NOR gate is a universal gate since it can implement the AND, OR and NOT functions.

Equivalent Gates

The shown figure summarizes important cases of gate equivalence. Note that bubbles indicate a complement operation (inverter).
A NAND gate is equivalent to an inverted-input OR gate.

An AND gate is equivalent to an inverted-input NOR gate.

A NOR gate is equivalent to an inverted-input AND gate

An OR gate is equivalent to an inverted-input NAND gate.

Two NOT gates in series are same as a buffer because they cancel each other as $A'' = A$. 

---

---
2.3 Working of an Exclusive - OR Gate with Truth Table

The output Q is true if either input A is true OR input B is true, but not when both of them are true: \[ Q = (A \ AND \ NOT \ B) \ OR \ (B \ AND \ NOT \ A) \]

This is like an OR gate but excluding both inputs being true.

The output is true if inputs A and B are DIFFERENT.

EX-OR gates can only have 2 inputs.

![Fig. 2.21 Two inputs Ex-OR Gate](image)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

![Fig. 2.22 Ex-OR Gate Truth Table](image)

2.4 State and Explain De-Morgan’s Theorems

A mathematician named DeMorgan developed a pair of important rules regarding group complementation in Boolean algebra.

It should be recalled from logic gates that inverting all inputs to a gate reverses that gate’s essential function from AND to OR, or vice versa, and also inverts the output. So, an OR gate with all inputs inverted (a Negative-OR gate) behaves the same as a NAND gate, and an AND gate with all inputs inverted (a Negative-AND gate) behaves the same as a NOR gate. DeMorgan’s theorems state the same equivalence in “backward” form: that inverting the output of any gate results in the same function as the opposite type of gate (AND vs. OR) with inverted inputs:
De-Morgan’s First Law: Product of the compliment is equal to Sum of the Compliments.

A long bar extending over the term AB acts as a grouping symbol, and as such is entirely different from the product of A and B independently inverted. In other words, (AB)' is not equal to A'B'. Because the “prime” symbol (’) cannot be stretched over two variables like a bar can, we are forced to use parentheses to make it apply to the whole term AB in the previous sentence. A bar, however, acts as its own grouping symbol when stretched over more than one variable. This has profound impact on how Boolean expressions are evaluated and reduced, as we shall see.

DeMorgan’s theorem may be thought of in terms of breaking a long bar symbol. When a long bar is broken, the operation directly underneath the break changes from addition to multiplication, or vice versa, and the broken bar pieces remain over the individual variables.

De-Morgan’s Second Law: Sum of the compliment is equal to Product of the Compliments.

When multiple “layers” of bars exist in an expression, you may only break one bar at a time, and it is generally easier to begin simplification by breaking the longest (uppermost) bar first. To illustrate, let’s take the expression (A + (BC)')' and reduce it using DeMorgan’s Theorems:
2.5 Develop AND, OR, NOT Operations using NAND, NOR Gates

Logic gates process signals which represent true or false. Normally the positive supply voltage $+V_s$ represents true and 0V represents false. Other terms which are used for the true and false states are shown in the table on the right. It is best to be familiar with them all.

Gates are identified by their function: NOT, AND, NAND, OR, NOR, EX-OR and EX-NOR. Capital letters are normally used to make it clear that the term refers to a logic gate.

Note that logic gates are not always required because simple logic functions
can be performed with switches or diodes:

· Switches in series (AND function)
· Switches in parallel (OR function)
· Combining IC outputs with diodes (OR function)

<table>
<thead>
<tr>
<th>LOGIC STATES</th>
<th>TRUE</th>
<th>FALSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>+Vs</td>
<td>OV</td>
<td>OV</td>
</tr>
<tr>
<td>On</td>
<td>OFF</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Fig. 2.25 Logic States

**Logic Gate Symbols**

There are two series of symbols for logic gates:

· The **Traditional symbols** have distinctive shapes making them easy to recognise so they are widely used in industry and education.

![Logic Symbols](image)

Fig. 2.26 Logic Symbols or AND, OR and NOT Gates

**Input and Outputs**

Gates have two or more inputs, except a NOT gate which has only one input. All gates have only one output. Usually the letters A, B, C and so on are used to label inputs, and Q is used to label the output. On this page the inputs are shown on the left and the output on the right.

![Input AND Gate](image)

Fig. 2.27 To Inputs AND Gates
The Inverting Circle (o)

Some gate symbols have a circle on their output which means that their function includes inverting of the output. It is equivalent to feeding the output through a NOT gate. For example the NAND (Not AND) gate symbol shown on the right is the same as an AND gate symbol but with the addition of an inverting circle on the output.

![Fig. 2.28 To Input NAND Gates](image)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 2.29 Truth Table AND Gate

A truth table is a good way to show the function of a logic gate. It shows the output states for every possible combination of input states. The symbols 0 (false) and 1 (true) are usually used in truth tables. The example truth table on the right shows the inputs and output of an AND gate.

There are summary truth tables below showing the output states for all types of 2-input and 3-input gates. These can be helpful if you are trying to select a suitable gate.

NOT Gate (Inverter)

The output Q is true when the input A is NOT true, the output is the inverse of the input: $Q = \text{NOT} A$

A NOT gate can only have one input. A NOT gate is also called an inverter.
In computer engineering, a **logic family** may refer to one of two related concepts. A logic family of monolithic digital integrated circuit devices is a group of electronic logic gates constructed using one of several different designs, usually with compatible logic levels and power supply characteristics within a family. Many logic families were produced as individual components, each containing one or a few related basic logical functions, which could be used as “building-blocks” to create systems or as so-called “glue” to interconnect more complex integrated circuits. A “logic family” may also refer to a set of techniques used to implement logic within VLSI integrated circuits such as central processors, memories, or other complex functions. Some such logic families use static techniques to minimize design complexity. Other such logic families, such as domino logic, use clocked dynamic techniques to minimize size, power consumption, and delay.

Before the widespread use of integrated circuits, various solid-state and vacuum-tube logic systems were used but these were never as standardized and interoperable as the integrated-circuit devices.

The list of packaged building-block logic families can be divided into categories, listed here in rough chronological order of introduction along with their usual abbreviations:

- Resistor–transistor logic (RTL)
  - Direct-coupled transistor logic (DCTL)
  - Resistor–capacitor–transistor logic (RCCTL)
- Diode–transistor logic (DTL)
  - Complemented transistor diode logic (CTDL)
  - High-threshold logic (HTL)
- Emitter-coupled logic (ECL)
· Positive emitter-coupled logic (PECL)
· Low-voltage positive emitter-coupled logic (LVPECL)
· Gunning transceiver logic (GTL)
· Transistor–transistor logic (TTL)
· P-type metal–oxide–semiconductor logic (PMOS)
· N-type metal–oxide–semiconductor logic (NMOS)
· Depletion-load NMOS logic
· Complementary metal–oxide–semiconductor logic (CMOS)
· Bipolar complementary metal–oxide–semiconductor logic (BiCMOS)
· Integrated injection logic (I\textsuperscript{2}L)

The families (RTL, DTL, and ECL) were derived from the logic circuits used in early computers, originally implemented using discrete components. One example is the Philips NORbits family of logic building blocks.

The PMOS and I\textsuperscript{2}L logic families were used for relatively short periods, mostly in special purpose custom LSI circuits devices and are generally considered obsolete. For example, early digital clocks or electronic calculators may have used one or more PMOS devices to provide most of the logic for the finished product. The F14 CADC, Intel 4004, Intel 4040, and Intel 8008 microprocessors and their support chips were PMOS.

Of these families, only ECL, TTL, NMOS, CMOS, and BiCMOS are currently still in widespread use. ECL is used for very high-speed applications because of its price and power demands, while NMOS logic is mainly used in VLSI circuits applications such as CPUs and memory chips which fall outside of the scope of this article. Present-day “building block” logic gate ICs are based on the ECL, TTL, CMOS, and BiCMOS families.

2.7 Characteristics of Digital IC’s Logic levels, Propagation Delay, Noise Margin, Fan In, Fan-Out and Power Dissipation

**Threshold Voltage** : It is the voltage at the input of the gate which causes a change in the state of the output from one logic level to the other.

**Propagation Delay** : It is the delay of a logic gate, the time taken by a pulse to propagate from input to output.
**Power Dissipation**: It is the power required by the logic gate to operate with 50% duty cycle at a specified frequency and is expressed in milli watts known as Power dissipation.

**Fan-In**: It is the number of inputs of a logic gate designed to handle known as Fan-In.

**Fan-Out**: It is the maximum number of similar gates that the output of the gate can drive its normal operations is known as Fan-Out.

**Noise Margin**: It is the maximum noise signal that can be added to the input signal of a digital circuit without causing an undesirable change in the circuit output is known as Noise Margin.

**Speed Power Product**: It is the product of the average propagation delay and the average power dissipation of a logic gate is known as speed power product.

### 2.8 Comparision of TTL, CMOS AND ECL Logic Families

Many logic families have been developed. They are Resistor Transistor Logic (RTL), Direct Coupled Transistor Logic (DCTL), Diode, Transistor Logic (DTL), High Threshold logic (HTL), Transistor Transistor Logic (TTL), Emitter Coupled Logic (ECL), Integrated Injection Logic (IIL), Metal oxide Semiconductor (MOS), and Complementary Metal oxide Semiconductor Logic (CMOS). Out of these, RTL, DCTL, and HTL are obsolete. The logic families TTL, ECL, IIL, MOS and CMOS are currently in use. The basic function of any type of gate is always the same regardless of the circuit technology used. The TTL and CMOS are suitable for SSI and MSI. The MOS and CMOS are particularly suitable for LSI. The IIL is mainly suitable for VLSI and ULSI. The ECL is mainly used in superfast computers. The logic families currently in use are compared in Table 2.1 in terms of the commonly used specification parameters

<table>
<thead>
<tr>
<th>Logic family</th>
<th>ns (ns)</th>
<th>mW (mW)</th>
<th>V (V)</th>
<th>Fan-in</th>
<th>Fan-out</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL</td>
<td>9</td>
<td>10</td>
<td>0.4</td>
<td>8</td>
<td>10</td>
<td>Low</td>
</tr>
<tr>
<td>ECL</td>
<td>1</td>
<td>50</td>
<td>0.25</td>
<td>5</td>
<td>10</td>
<td>High</td>
</tr>
<tr>
<td>MOS</td>
<td>50</td>
<td>0.1</td>
<td>1.5</td>
<td>10</td>
<td></td>
<td>Low</td>
</tr>
<tr>
<td>CMOS</td>
<td>&lt;50</td>
<td>0.01</td>
<td>5</td>
<td>10</td>
<td>50</td>
<td>Low</td>
</tr>
<tr>
<td>IIL</td>
<td>1</td>
<td>0.1</td>
<td>0.35</td>
<td>5</td>
<td>8</td>
<td>Very low</td>
</tr>
</tbody>
</table>

ns = Propagation delay time  mW = Power dissipation per gate
### 2.9 Specification and Number of Digital IC Logic Gates

The following is a list of 7400 series digital logic integrated circuits. The SN7400 series originated with TTL integrated circuits made by Texas Instruments. Because of the popularity of these parts, they were second-sourced by other manufacturers who kept the 7400 sequence number as an aid to identification of compatible parts. As well, compatible TTL parts originated by other manufacturers were second sourced in the TI product line under a 74xxx series part number.

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7400</td>
<td>quad 2-input NAND gate</td>
</tr>
<tr>
<td>741G00</td>
<td>single 2-input NAND gate</td>
</tr>
<tr>
<td>7401</td>
<td>quad 2-input NAND gate with open collector outputs</td>
</tr>
<tr>
<td>741G01</td>
<td>single 2-input NAND gate with open drain output</td>
</tr>
<tr>
<td>7402</td>
<td>quad 2-input NOR gate</td>
</tr>
<tr>
<td>741G02</td>
<td>single 2-input NOR gate</td>
</tr>
<tr>
<td>7403</td>
<td>quad 2-input NAND gate with open collector outputs</td>
</tr>
<tr>
<td>741G03</td>
<td>single 2-input NAND gate with open drain output</td>
</tr>
<tr>
<td>7404</td>
<td>hex inverter</td>
</tr>
<tr>
<td>741G04</td>
<td>single inverter</td>
</tr>
<tr>
<td>7405</td>
<td>hex inverter with open collector outputs</td>
</tr>
<tr>
<td>741G05</td>
<td>single inverter with open drain output</td>
</tr>
<tr>
<td>7406</td>
<td>hex inverter buffer/driver with 30 v open collector outputs</td>
</tr>
<tr>
<td>741G06</td>
<td>single inverting buffer/driver with open drain output</td>
</tr>
<tr>
<td>7407</td>
<td>hex buffer/driver with 30 v open collector outputs</td>
</tr>
<tr>
<td>741G07</td>
<td>single non-inverting buffer/driver with open drain output</td>
</tr>
<tr>
<td>7408</td>
<td>quad 2-input AND gate</td>
</tr>
<tr>
<td>741G08</td>
<td>single 2-input AND gate</td>
</tr>
<tr>
<td>7409</td>
<td>quad 2-input AND gate with open collector outputs</td>
</tr>
<tr>
<td>741G09</td>
<td>single 2-input AND gate with open drain output</td>
</tr>
<tr>
<td>7410</td>
<td>triple 3-input NAND gate</td>
</tr>
<tr>
<td>7411</td>
<td>triple 3-input AND gate</td>
</tr>
<tr>
<td>7412</td>
<td>triple 3-input NAND gate with open collector outputs</td>
</tr>
<tr>
<td>7413</td>
<td>dual Schmitt trigger 4-input NAND gate</td>
</tr>
<tr>
<td>7414</td>
<td>hex Schmitt trigger inverter</td>
</tr>
<tr>
<td>741G14</td>
<td>single Schmitt trigger inverter</td>
</tr>
<tr>
<td>7415</td>
<td>triple 3-input AND gate with open collector outputs</td>
</tr>
<tr>
<td>7416</td>
<td>hex inverter buffer/driver with 15 v open collector outputs</td>
</tr>
<tr>
<td>7417</td>
<td>hex buffer/driver with 15 v open collector outputs</td>
</tr>
<tr>
<td>741G17</td>
<td>single Schmitt-trigger buffer</td>
</tr>
</tbody>
</table>
7418  dual 4-input NAND gate with Schmitt trigger inputs
7419  hex Schmitt trigger inverter
7420  dual 4-input NAND gate
7421  dual 4-input AND gate
7422  dual 4-input NAND gate with open collector outputs
7423  expandable dual 4-input NOR gate with strobe
7424  quad 2-input NAND gate gates with schmitt-trigger line-receiver inputs.
7425  dual 4-input NOR gate with strobe
7426  quad 2-input NAND gate with 15 v open collector outputs
7427  triple 3-input NOR gate
741G27  single 3-input NOR gate
7428  quad 2-input NOR buffer
7430  8-input NAND gate
7431  hex delay elements
7432  quad 2-input OR gate
741G32  single 2-input OR gate
7433  quad 2-input NOR buffer with open collector outputs
7436  quad 2-input NOR gate (different pinout than 7402)
7437  quad 2-input NAND buffer
7438  quad 2-input NAND buffer with open collector outputs
7439  quad 2-input NAND buffer
7440  dual 4-input NAND buffer
7441  BCD to decimal decoder/Nixie tube driver
7442  BCD to decimal decoder
7443  excess-3 to decimal decoder
7444  excess-3-Gray code to decimal decoder
7445  BCD to decimal decoder/driver
7446  BCD to seven-segment display decoder/driver with 30 v open collector outputs
7447  BCD to 7-segment decoder/driver with 15 v open collector outputs
7448  BCD to 7-segment decoder/driver with Internal Pullups
7449  BCD to 7-segment decoder/driver with open collector outputs
7450  dual 2-wide 2-input AND-OR-invert gate (one gate expandable)
7451  dual 2-wide 2-input AND-OR-invert gate
7452  expandable 4-wide 2-input AND-OR gate
<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7453</td>
<td>expandable 4-wide 2-input AND-OR-invert gate</td>
</tr>
<tr>
<td>7454</td>
<td>4-wide 2-input AND-OR-invert gate</td>
</tr>
<tr>
<td>7455</td>
<td>2-wide 4-input AND-OR-invert Gate (74H version is expandable)</td>
</tr>
<tr>
<td>7456</td>
<td>50:1 frequency divider</td>
</tr>
<tr>
<td>7457</td>
<td>60:1 frequency divider</td>
</tr>
<tr>
<td>7458</td>
<td>2-input &amp; 3-input AND-OR Gate</td>
</tr>
<tr>
<td>7459</td>
<td>2-input &amp; 3-input AND-OR-invert Gate</td>
</tr>
<tr>
<td>7460</td>
<td>dual 4-input expander</td>
</tr>
<tr>
<td>7461</td>
<td>triple 3-input expander</td>
</tr>
<tr>
<td>7462</td>
<td>3-2-2-3-input AND-OR expander</td>
</tr>
<tr>
<td>7463</td>
<td>hex current sensing interface gates</td>
</tr>
<tr>
<td>7464</td>
<td>4-2-3-2-input AND-OR-invert gate</td>
</tr>
<tr>
<td>7465</td>
<td>4-2-3-2 input AND-OR-invert gate with open collector output</td>
</tr>
<tr>
<td>7468</td>
<td>dual 4 bit decade counters</td>
</tr>
<tr>
<td>7469</td>
<td>dual 4 bit binary counters</td>
</tr>
<tr>
<td>7470</td>
<td>AND-gated positive edge triggered J-K flip-flop with preset and clear</td>
</tr>
<tr>
<td>74H71</td>
<td>AND-or-gated J-K master-slave flip-flop with preset</td>
</tr>
<tr>
<td>74L71</td>
<td>AND-gated R-S master-slave flip-flop with preset and clear</td>
</tr>
<tr>
<td>7472</td>
<td>AND gated J-K master-slave flip-flop with preset and clear</td>
</tr>
<tr>
<td>7473</td>
<td>dual J-K flip-flop with clear</td>
</tr>
<tr>
<td>7474</td>
<td>dual D positive edge triggered flip-flop with preset and clear</td>
</tr>
<tr>
<td>7475</td>
<td>4-bit bistable latch</td>
</tr>
<tr>
<td>7476</td>
<td>dual J-K flip-flop with preset and clear</td>
</tr>
<tr>
<td>7477</td>
<td>4-bit bistable latch</td>
</tr>
<tr>
<td>74H78</td>
<td>dual positive pulse triggered J-K flip-flop with preset, common clock, and common clear (different pinout than 74L78 / 74Ls78)</td>
</tr>
<tr>
<td>74L78</td>
<td>dual positive pulse triggered J-K flip-flop with preset, common clock, and common clear</td>
</tr>
<tr>
<td>74Ls78</td>
<td>dual negative edge triggered J-K flip-flop with preset, common clock, and common clear</td>
</tr>
<tr>
<td>7479</td>
<td>dual D flip-flop</td>
</tr>
<tr>
<td>741G79</td>
<td>single D-type flip-flop positive edge trigger non-inverting output</td>
</tr>
<tr>
<td>7480</td>
<td>gated full adder</td>
</tr>
<tr>
<td>741G80</td>
<td>single D-type flip-flop positive edge trigger inverting output</td>
</tr>
<tr>
<td>7481</td>
<td>16-bit random access memory</td>
</tr>
<tr>
<td>7482</td>
<td>2-bit binary full adder</td>
</tr>
<tr>
<td>7483</td>
<td>4-bit binary full adder</td>
</tr>
<tr>
<td>7484</td>
<td>16-bit random access memory</td>
</tr>
<tr>
<td>7485</td>
<td>4-bit magnitude comparator</td>
</tr>
<tr>
<td>7486</td>
<td>quad 2-input XOR gate</td>
</tr>
<tr>
<td>741G86</td>
<td>single 2-input exclusive-OR gate</td>
</tr>
<tr>
<td>7487</td>
<td>4-bit true/complement/zero/one element</td>
</tr>
<tr>
<td>7488</td>
<td>256-bit read-only memory</td>
</tr>
<tr>
<td>7489</td>
<td>64-bit random access memory</td>
</tr>
<tr>
<td>7490</td>
<td>decade counter (separate divide-by-2 and divide-by-5 sections)</td>
</tr>
<tr>
<td>7491</td>
<td>8-bit shift register, serial In, serial out, gated input</td>
</tr>
<tr>
<td>7492</td>
<td>divide-by-12 counter (separate divide-by-2 and divide-by-6 sections)</td>
</tr>
<tr>
<td>7493</td>
<td>4-bit binary counter (separate divide-by-2 and divide-by-8 sections)</td>
</tr>
<tr>
<td>7494</td>
<td>4-bit shift register, dual asynchronous presets</td>
</tr>
<tr>
<td>7495</td>
<td>4-bit shift register, parallel In, parallel out, serial input</td>
</tr>
<tr>
<td>7496</td>
<td>5-bit parallel-In/parallel-out shift register, asynchronous preset</td>
</tr>
<tr>
<td>7497</td>
<td>synchronous 6-bit binary rate multiplier</td>
</tr>
<tr>
<td>741G97</td>
<td>configurable multiple-function gate</td>
</tr>
<tr>
<td>7498</td>
<td>4-bit data selector/storage register</td>
</tr>
<tr>
<td>7499</td>
<td>4-bit bidirectional universal shift register</td>
</tr>
<tr>
<td>74100</td>
<td>dual 4-bit bistable latch</td>
</tr>
<tr>
<td>74101</td>
<td>AND-or-gated J-K negative-edge-triggered flip-flop with preset</td>
</tr>
<tr>
<td>74102</td>
<td>AND-gated J-K negative-edge-triggered flip-flop with preset and clear</td>
</tr>
<tr>
<td>74103</td>
<td>dual J-K negative-edge-triggered flip-flop with clear</td>
</tr>
<tr>
<td>74104</td>
<td>J-K master-slave flip-flop</td>
</tr>
<tr>
<td>74105</td>
<td>J-K master-slave flip-flop</td>
</tr>
<tr>
<td>74106</td>
<td>dual J-K negative-edge-triggered flip-flop with preset and clear</td>
</tr>
<tr>
<td>74107</td>
<td>dual J-K flip-flop with clear</td>
</tr>
<tr>
<td>74107a</td>
<td>dual J-K negative-edge-triggered flip-flop with clear</td>
</tr>
<tr>
<td>74108</td>
<td>dual J-K negative-edge-triggered flip-flop with preset,</td>
</tr>
</tbody>
</table>
74109 common clear, and common clock
dual J-Not-K positive-edge-triggered flip-flop with clear and preset
74110 AND-gated J-K master-slave flip-flop with data lockout
74111 dual J-K master-slave flip-flop with data lockout
74112 dual J-K negative-edge-triggered flip-flop with clear and preset
74113 dual J-K negative-edge-triggered flip-flop with preset
74114 dual J-K negative-edge-triggered flip-flop with preset, common clock and clear
74116 dual 4-bit latch with clear
74118 hex set/reset latch
74120 dual pulse synchronizer/drivers
74121 monostable multivibrator
74122 retriggerable monostable multivibrator with clear
74123 dual retriggerable monostable multivibrator with clear
741G123 single retriggerable monostable multivibrator with clear
74124 dual voltage-controlled oscillator
74125 quad bus buffer with three-state outputs, negative enable
741G125 buffer/Line driver, three-state output with active low output enable
74126 quad bus buffer with three-state outputs, positive enable
74128 quad 2-input NOR Line driver
741G126 buffer/line driver, three-state output with active high output enable
74130 quad 2-input AND gate buffer with 30 v open collector outputs
74131 quad 2-input AND gate buffer with 15 v open collector outputs
74132 quad 2-input NAND schmitt trigger
74133 13-input NAND gate
74134 12-input NAND gate with three-state output
74135 quad exclusive-or/NOR gate
74136 quad 2-input XOR gate with open collector outputs
74137 3 to 8-line decoder/demultiplexer with address latch
74138 3 to 8-line decoder/demultiplexer
74139 dual 2 to 4-line decoder/demultiplexer
74140 dual 4-input NAND line driver
<table>
<thead>
<tr>
<th>74141</th>
<th>BCD to decimal decoder/driver for cold-cathode indicator/Nixie tube</th>
</tr>
</thead>
<tbody>
<tr>
<td>74142</td>
<td>decade counter/latch/decoder/driver for Nixie tubes</td>
</tr>
<tr>
<td>74143</td>
<td>decade counter/latch/decoder/7-segment driver, 15 ma constant current</td>
</tr>
<tr>
<td>74144</td>
<td>decade counter/latch/decoder/7-segment driver, 15 v open collector outputs</td>
</tr>
<tr>
<td>74145</td>
<td>BCD to decimal decoder/driver</td>
</tr>
<tr>
<td>74147</td>
<td>10-line to 4-line priority encoder</td>
</tr>
<tr>
<td>74148</td>
<td>8-line to 3-line priority encoder</td>
</tr>
<tr>
<td>74150</td>
<td>16-line to 1-line data selector/multiplexer</td>
</tr>
<tr>
<td>74151</td>
<td>8-line to 1-line data selector/multiplexer</td>
</tr>
<tr>
<td>74152</td>
<td>8-line to 1-line data selector/multiplexer</td>
</tr>
<tr>
<td>74153</td>
<td>dual 4-line to 1-line data selector/multiplexer</td>
</tr>
<tr>
<td>74154</td>
<td>4-line to 16-line decoder/demultiplexer</td>
</tr>
<tr>
<td>74155</td>
<td>dual 2-line to 4-line decoder/demultiplexer</td>
</tr>
<tr>
<td>74156</td>
<td>dual 2-line to 4-line decoder/demultiplexer with open collector outputs</td>
</tr>
<tr>
<td>74157</td>
<td>quad 2-line to 1-line data selector/multiplexer, noninverting</td>
</tr>
<tr>
<td>74158</td>
<td>quad 2-line to 1-line data selector/multiplexer, inverting</td>
</tr>
<tr>
<td>74159</td>
<td>4-line to 16-line decoder/demultiplexer with open collector outputs</td>
</tr>
<tr>
<td>74160</td>
<td>synchronous 4-bit decade counter with asynchronous clear</td>
</tr>
<tr>
<td>74161</td>
<td>synchronous 4-bit binary counter with asynchronous clear</td>
</tr>
<tr>
<td>74162</td>
<td>synchronous 4-bit decade counter with synchronous clear</td>
</tr>
<tr>
<td>74163</td>
<td>synchronous 4-bit binary counter with synchronous clear</td>
</tr>
<tr>
<td>74164</td>
<td>8-bit parallel-out serial shift register with asynchronous clear</td>
</tr>
<tr>
<td>74165</td>
<td>8-bit serial shift register, Parallel Load, complementary outputs</td>
</tr>
<tr>
<td>74166</td>
<td>parallel-Load 8-bit shift register</td>
</tr>
<tr>
<td>74167</td>
<td>synchronous decade rate multiplier</td>
</tr>
<tr>
<td>74168</td>
<td>synchronous 4-bit up/down decade counter</td>
</tr>
<tr>
<td>74169</td>
<td>synchronous 4-bit up/down binary counter</td>
</tr>
<tr>
<td>74170</td>
<td>4 by 4 register file with open collector outputs</td>
</tr>
<tr>
<td>74172</td>
<td>16-bit multiple port register file with three-state outputs</td>
</tr>
<tr>
<td>74173</td>
<td>quad d flip-flop with three-state outputs</td>
</tr>
<tr>
<td>74174</td>
<td>hex d flip-flop with common clear</td>
</tr>
<tr>
<td>74175</td>
<td>quad d edge-triggered flip-flop with complementary outputs and asynchronous clear</td>
</tr>
<tr>
<td>74176</td>
<td>presettable decade (bi-quinary) counter/latch</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>74177</td>
<td>presettable binary counter/latch</td>
</tr>
<tr>
<td>74178</td>
<td>4-bit parallel-access shift register</td>
</tr>
<tr>
<td>74179</td>
<td>4-bit parallel-access shift register with asynchronous clear and complementary Qd outputs</td>
</tr>
<tr>
<td>74180</td>
<td>9-bit odd/even parity bit generator and checker</td>
</tr>
<tr>
<td>74181</td>
<td>4-bit arithmetic logic unit and function generator</td>
</tr>
<tr>
<td>74182</td>
<td>lookahead carry generator</td>
</tr>
<tr>
<td>74183</td>
<td>dual carry-save full adder</td>
</tr>
<tr>
<td>74184</td>
<td>BCD to binary converter</td>
</tr>
<tr>
<td>74185</td>
<td>6-bit binary to BCD converter</td>
</tr>
<tr>
<td>74186</td>
<td>512-bit (64x8) read-only memory with open collector outputs</td>
</tr>
<tr>
<td>74187</td>
<td>1024-bit (256x4) read only memory with open collector outputs</td>
</tr>
<tr>
<td>74188</td>
<td>256-bit (32x8) programmable read-only memory with open collector outputs</td>
</tr>
<tr>
<td>74189</td>
<td>64-bit (16x4) ram with inverting three-state outputs</td>
</tr>
<tr>
<td>74190</td>
<td>synchronous up/down decade counter</td>
</tr>
<tr>
<td>74191</td>
<td>synchronous up/down binary counter</td>
</tr>
<tr>
<td>74192</td>
<td>synchronous up/down decade counter with clear</td>
</tr>
<tr>
<td>74193</td>
<td>synchronous up/down binary counter with clear</td>
</tr>
<tr>
<td>74194</td>
<td>4-bit bidirectional universal shift register</td>
</tr>
<tr>
<td>74195</td>
<td>4-bit parallel-access shift register</td>
</tr>
<tr>
<td>74196</td>
<td>presettable decade counter/latch</td>
</tr>
<tr>
<td>74197</td>
<td>presettable binary counter/latch</td>
</tr>
<tr>
<td>74198</td>
<td>8-bit bidirectional universal shift register</td>
</tr>
<tr>
<td>74199</td>
<td>8-bit bidirectional universal shift register with J-Not-K serial inputs</td>
</tr>
<tr>
<td>74200</td>
<td>256-bit ram with three-state outputs</td>
</tr>
<tr>
<td>74201</td>
<td>256-bit (256x1) ram with three-state outputs</td>
</tr>
<tr>
<td>74206</td>
<td>256-bit ram with open collector outputs</td>
</tr>
<tr>
<td>74209</td>
<td>1024-bit (1024x1) ram with three-state output</td>
</tr>
<tr>
<td>74210</td>
<td>octal buffer</td>
</tr>
<tr>
<td>74219</td>
<td>64-bit (16x4) ram with noninverting three-state outputs</td>
</tr>
<tr>
<td>74221</td>
<td>dual monostable multivibrator with schmitt trigger input</td>
</tr>
<tr>
<td>74222</td>
<td>16 by 4 synchronous FIFO memory with three-state outputs</td>
</tr>
<tr>
<td>74224</td>
<td>16 by 4 synchronous FIFO memory with three-state</td>
</tr>
<tr>
<td>74225</td>
<td>asynchronous 16x5 FIFO memory</td>
</tr>
<tr>
<td>74226</td>
<td>4-bit parallel latched bus transceiver with three-state outputs</td>
</tr>
<tr>
<td>74230</td>
<td>octal buffer/driver with three-state outputs</td>
</tr>
<tr>
<td>74232</td>
<td>quad NOR Schmitt trigger</td>
</tr>
<tr>
<td>74237</td>
<td>1-of-8 decoder/demultiplexer with address latch, active high outputs</td>
</tr>
<tr>
<td>74238</td>
<td>1-of-8 decoder/demultiplexer, active high outputs</td>
</tr>
<tr>
<td>74239</td>
<td>dual 2-of-4 decoder/demultiplexer, active high outputs</td>
</tr>
<tr>
<td>74240</td>
<td>octal buffer with Inverted three-state outputs</td>
</tr>
<tr>
<td>74241</td>
<td>octal buffer with noninverted three-state outputs</td>
</tr>
<tr>
<td>74242</td>
<td>quad bus transceiver with Inverted three-state outputs</td>
</tr>
<tr>
<td>74243</td>
<td>quad bus transceiver with noninverted three-state outputs</td>
</tr>
<tr>
<td>74244</td>
<td>octal buffer with noninverted three-state outputs</td>
</tr>
<tr>
<td>74245</td>
<td>octal bus transceiver with noninverted three-state outputs</td>
</tr>
<tr>
<td>74246</td>
<td>BCD to 7-segment decoder/driver with 30 v open collector outputs</td>
</tr>
<tr>
<td>74247</td>
<td>BCD to 7-segment decoder/driver with 15 v open collector outputs</td>
</tr>
<tr>
<td>74248</td>
<td>BCD to 7-segment decoder/driver with Internal Pull-up outputs</td>
</tr>
<tr>
<td>74249</td>
<td>BCD to 7-segment decoder/driver with open collector outputs</td>
</tr>
<tr>
<td>74251</td>
<td>8-line to 1-line data selector/multiplexer with complementary three-state outputs</td>
</tr>
<tr>
<td>74253</td>
<td>dual 4-line to 1-line data selector/multiplexer with three-state outputs</td>
</tr>
<tr>
<td>74255</td>
<td>dual 4-bit addressable latch</td>
</tr>
<tr>
<td>74256</td>
<td>dual 4-bit addressable latch</td>
</tr>
<tr>
<td>74257</td>
<td>quad 2-line to 1-line data selector/multiplexer with noninverted three-state outputs</td>
</tr>
<tr>
<td>74258</td>
<td>quad 2-line to 1-line data selector/multiplexer with Inverted three-state outputs</td>
</tr>
<tr>
<td>74259</td>
<td>8-bit addressable latch</td>
</tr>
<tr>
<td>74260</td>
<td>dual 5-input NOR gate</td>
</tr>
<tr>
<td>74261</td>
<td>2-bit by 4-bit parallel binary multiplier</td>
</tr>
<tr>
<td>74265</td>
<td>quad complementary output elements</td>
</tr>
<tr>
<td>74266</td>
<td>quad 2-input XNOR gate with open collector outputs</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>74270</td>
<td>2048-bit (512x4) read only memory with open collector outputs</td>
</tr>
<tr>
<td>74271</td>
<td>2048-bit (256x8) read only memory with open collector outputs</td>
</tr>
<tr>
<td>74273</td>
<td>8-bit register with reset</td>
</tr>
<tr>
<td>74274</td>
<td>4-bit by 4-bit binary multiplier</td>
</tr>
<tr>
<td>74275</td>
<td>7-bit slice Wallace tree</td>
</tr>
<tr>
<td>74276</td>
<td>quad J-Not-K edge-triggered Flip-Flops with separate clocks, common preset and clear</td>
</tr>
<tr>
<td>74278</td>
<td>4-bit cascadeable priority registers with latched data inputs</td>
</tr>
<tr>
<td>74279</td>
<td>quad set-reset latch</td>
</tr>
<tr>
<td>74280</td>
<td>9-bit odd/even Parity bit Generator/checker</td>
</tr>
<tr>
<td>74281</td>
<td>4-bit parallel binary accumulator</td>
</tr>
<tr>
<td>74283</td>
<td>4-bit binary Full adder</td>
</tr>
<tr>
<td>74284</td>
<td>4-bit by 4-bit parallel binary multiplier (low order 4 bits of product)</td>
</tr>
<tr>
<td>74285</td>
<td>4-bit by 4-bit parallel binary multiplier (high order 4 bits of product)</td>
</tr>
<tr>
<td>74287</td>
<td>1024-bit (256x4) programmable read-only memory with three-state outputs</td>
</tr>
<tr>
<td>74288</td>
<td>256-bit (32x8) programmable read-only memory with three-state outputs</td>
</tr>
<tr>
<td>74289</td>
<td>64-bit (16x4) RAM with open collector outputs</td>
</tr>
<tr>
<td>74290</td>
<td>decade counter (separate divide-by-2 and divide-by-5 sections)</td>
</tr>
<tr>
<td>74291</td>
<td>4-bit universal shift register, binary up/down counter, synchronous</td>
</tr>
<tr>
<td>74292</td>
<td>programmable frequency divider/digital timer</td>
</tr>
<tr>
<td>74293</td>
<td>4-bit binary counter (separate divide-by-2 and divide-by-8 sections)</td>
</tr>
<tr>
<td>74294</td>
<td>programmable frequency divider/digital timer</td>
</tr>
<tr>
<td>74295</td>
<td>4-bit bidirectional register with three-state outputs</td>
</tr>
<tr>
<td>74297</td>
<td>digital phase-locked-loop filter</td>
</tr>
<tr>
<td>74298</td>
<td>quad 2-input multiplexer with storage</td>
</tr>
<tr>
<td>74299</td>
<td>8-bit bidirectional universal shift/storage register with three-state outputs</td>
</tr>
<tr>
<td>74301</td>
<td>256-bit (256x1) random access memory with open collector output</td>
</tr>
<tr>
<td>74309</td>
<td>1024-bit (1024x1) random access memory with open</td>
</tr>
</tbody>
</table>
Electronic Engineering Technician

74310  collector output
74310  octal buffer with Schmitt trigger inputs
74314  1024-bit random access memory
74320  crystal controlled oscillator
74322  8-bit shift register with sign extend, three-state outputs
74323  8-bit bidirectional universal shift/storage register with three-state outputs
74324  voltage controlled oscillator (or crystal controlled)
74340  octal buffer with Schmitt trigger inputs and three-state inverted outputs
74341  octal buffer with Schmitt trigger inputs and three-state noninverted outputs
74344  octal buffer with Schmitt trigger inputs and three-state noninverted outputs
74348  8 to 3-line priority encoder with three-state outputs
74350  4-bit shifter with three-state outputs
74351  dual 8-line to 1-line data selectors/multiplexers with three-state outputs and 4 common data inputs
74352  dual 4-line to 1-line data selectors/multiplexers with inverting outputs
74353  dual 4-line to 1-line data selectors/multiplexers with inverting three-state outputs
74354  8 to 1-line data selector/multiplexer with transparent latch, three-state outputs
74356  8 to 1-line data selector/multiplexer with edge-triggered register, three-state outputs
74361  bubble memory function timing generator
74362  four-phase clock generator/driver
74365  hex buffer with noninverted three-state outputs
74366  hex buffer with inverted three-state outputs
74367  hex buffer with noninverted three-state outputs
74368  hex buffer with inverted three-state outputs
74370  2048-bit (512x4) read-only memory with three-state outputs
74371  2048-bit (256x8) read-only memory with three-state outputs
74373  octal transparent latch with three-state outputs
741G373  single transparent latch with three-state output
74374  octal register with three-state outputs
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>741G374</td>
<td>single d-type flip-flop with three-state output</td>
</tr>
<tr>
<td>74375</td>
<td>quad bistable latch</td>
</tr>
<tr>
<td>74376</td>
<td>quad J-Not-K flip-flop with common clock and common clear</td>
</tr>
<tr>
<td>74377</td>
<td>8-bit register with clock enable</td>
</tr>
<tr>
<td>74378</td>
<td>6-bit register with clock enable</td>
</tr>
<tr>
<td>74379</td>
<td>4-bit register with clock enable and complementary outputs</td>
</tr>
<tr>
<td>74380</td>
<td>8-bit multifunction register</td>
</tr>
<tr>
<td>74381</td>
<td>4-bit arithmetic logic unit/function generator with generate and propagate outputs</td>
</tr>
<tr>
<td>74382</td>
<td>4-bit arithmetic logic unit/function generator with ripple carry and overflow outputs</td>
</tr>
<tr>
<td>74385</td>
<td>quad 4-bit adder/subtractor</td>
</tr>
<tr>
<td>74386</td>
<td>quad 2-input XOR gate</td>
</tr>
<tr>
<td>74387</td>
<td>1024-bit (256x4) programmable read-only memory with open collector outputs</td>
</tr>
<tr>
<td>74388</td>
<td>4-bit register with standard and three-state outputs</td>
</tr>
<tr>
<td>74390</td>
<td>dual 4-bit decade counter</td>
</tr>
<tr>
<td>74393</td>
<td>dual 4-bit binary counter</td>
</tr>
<tr>
<td>74395</td>
<td>4-bit universal shift register with three-state outputs</td>
</tr>
<tr>
<td>74398</td>
<td>quad 2-input multiplexers with storage and complementary outputs</td>
</tr>
<tr>
<td>74399</td>
<td>quad 2-input multiplexer with storage</td>
</tr>
<tr>
<td>74405</td>
<td>1 to 8 decoder, equivalent to Intel 8205, only found as UCY74S405 so might be non-TI number</td>
</tr>
<tr>
<td>74408</td>
<td>8-bit parity tree</td>
</tr>
<tr>
<td>74412</td>
<td>multi-mode buffered 8-bit latches with three-state outputs and clear</td>
</tr>
<tr>
<td>74423</td>
<td>dual retriggerable monostable multivibrator</td>
</tr>
<tr>
<td>74424</td>
<td>two-phase clock generator/driver</td>
</tr>
<tr>
<td>74425</td>
<td>quad gates with three-state outputs and active low enables</td>
</tr>
<tr>
<td>74426</td>
<td>quad gates with three-state outputs and active high enables</td>
</tr>
<tr>
<td>74428</td>
<td>system controller for 8080a</td>
</tr>
<tr>
<td>74438</td>
<td>system controller for 8080a</td>
</tr>
<tr>
<td>74440</td>
<td>quad tridirectional bus transceiver with noninverted open collector outputs</td>
</tr>
<tr>
<td>74441</td>
<td>quad tridirectional bus transceiver with Inverted open collector outputs</td>
</tr>
</tbody>
</table>
| 74442       | quad tridirectional bus transceiver with noninverted three-
state outputs
74443 quad tridirectional bus transceiver with Inverted three-state outputs
74444 quad tridirectional bus transceiver with Inverted and noninverted three-state outputs
74448 quad tridirectional bus transceiver with Inverted and noninverted open collector outputs
74450 16-to-1 multiplexer with complementary outputs
74451 dual 8-to-1 multiplexer
74452 dual decade counter, synchronous
74453 dual binary counter, synchronous
74453 quad 4-to-1 multiplexer
74454 dual decade up/down counter, synchronous, preset input
74455 dual binary up/down counter, synchronous, preset input
74456 NBCD (Natural binary coded decimal) adder
74460 bus transfer switch
74461 8-bit presettable binary counter with three-state outputs
74462 fiber-optic link transmitter
74463 fiber-optic link receiver
74465 octal buffer with three-state outputs
74468 dual mos-to-ttl level converter
74470 2048-bit (256x8) programmable read-only memory with open collector outputs
74471 2048-bit (256x8) programmable read-only memory with three-state outputs
74472 programmable read-only memory with open collector outputs
74473 programmable read-only memory with three-state outputs
74474 programmable read-only memory with open collector outputs
74475 programmable read-only memory with three-state outputs
74481 4-bit slice processor elements
74482 4-bit slice expandable control elements
74484 BCD-to-binary converter
74485 binary-to-BCD converter
74490 dual decade counter
74491 10-bit binary up/down counter with limited preset and three-state outputs
74498 8-bit bidirectional shift register with parallel inputs and
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>74508</td>
<td>8-bit multiplier/divider</td>
</tr>
<tr>
<td>74520</td>
<td>8-bit comparator</td>
</tr>
<tr>
<td>74521</td>
<td>8-bit comparator</td>
</tr>
<tr>
<td>74526</td>
<td>fuse programmable identity comparator, 16 bit</td>
</tr>
<tr>
<td>74527</td>
<td>fuse programmable identity comparator, 8 bit + 4 bit</td>
</tr>
<tr>
<td></td>
<td>conventional Identity comparator</td>
</tr>
<tr>
<td>74528</td>
<td>fuse programmable Identity comparator, 12 bit</td>
</tr>
<tr>
<td>74531</td>
<td>octal transparent latch with 32 ma three-state outputs</td>
</tr>
<tr>
<td>74532</td>
<td>octal register with 32 ma three-state outputs</td>
</tr>
<tr>
<td>74533</td>
<td>octal transparent latch with inverting three-state Logic outputs</td>
</tr>
<tr>
<td></td>
<td>octal register with inverting three-state outputs</td>
</tr>
<tr>
<td>74535</td>
<td>octal transparent latch with inverting three-state outputs</td>
</tr>
<tr>
<td>74536</td>
<td>octal register with inverting 32 ma three-state outputs</td>
</tr>
<tr>
<td>74537</td>
<td>BCD to decimal decoder with three-state outputs</td>
</tr>
<tr>
<td>74538</td>
<td>1 of 8 decoder with three-state outputs</td>
</tr>
<tr>
<td>74539</td>
<td>dual 1 of 4 decoder with three-state outputs</td>
</tr>
<tr>
<td>74540</td>
<td>inverting octal buffer with three-state outputs</td>
</tr>
<tr>
<td>74541</td>
<td>non-inverting octal buffer with three-state outputs</td>
</tr>
<tr>
<td>74544</td>
<td>non-inverting octal registered transceiver with three-state outputs</td>
</tr>
<tr>
<td>74558</td>
<td>8-bit by 8-bit multiplier with three-state outputs</td>
</tr>
<tr>
<td>74560</td>
<td>4-bit decade counter with three-state outputs</td>
</tr>
<tr>
<td>74561</td>
<td>4-bit binary counter with three-state outputs</td>
</tr>
<tr>
<td>74563</td>
<td>8-bit d-type transparent latch with inverting three-state outputs</td>
</tr>
<tr>
<td>74564</td>
<td>8-bit d-type edge-triggered register with inverting three-state outputs</td>
</tr>
<tr>
<td>74568</td>
<td>decade up/down counter with three-state outputs</td>
</tr>
<tr>
<td>74569</td>
<td>binary up/down counter with three-state outputs</td>
</tr>
<tr>
<td>74573</td>
<td>octal D-type transparent latch with three-state outputs</td>
</tr>
<tr>
<td>74574</td>
<td>octal D-type edge-triggered flip-flop with three-state outputs</td>
</tr>
<tr>
<td>74575</td>
<td>octal D-type flip-flop with synchronous clear, three-state outputs</td>
</tr>
<tr>
<td>74576</td>
<td>octal D-type flip-flop with inverting three-state outputs</td>
</tr>
<tr>
<td>74577</td>
<td>octal D-type flip-flop with synchronous clear, inverting three-state outputs</td>
</tr>
<tr>
<td>Number</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>74580</td>
<td>octal transceiver/latch with inverting three-state outputs</td>
</tr>
<tr>
<td>74589</td>
<td>8-bit shift register with input latch, three-state outputs</td>
</tr>
<tr>
<td>74590</td>
<td>8-bit binary counter with output registers and three-state outputs</td>
</tr>
<tr>
<td>74592</td>
<td>8-bit binary counter with input registers</td>
</tr>
<tr>
<td>74593</td>
<td>8-bit binary counter with input registers and three-state outputs</td>
</tr>
<tr>
<td>74594</td>
<td>serial-in shift register with output registers</td>
</tr>
<tr>
<td>74595</td>
<td>serial-in shift register with output latches</td>
</tr>
<tr>
<td>74596</td>
<td>serial-in shift register with output registers and open collector outputs</td>
</tr>
<tr>
<td>74597</td>
<td>serial-out shift register with input latches</td>
</tr>
<tr>
<td>74598</td>
<td>shift register with input latches</td>
</tr>
<tr>
<td>74600</td>
<td>dynamic memory refresh controller, transparent and burst modes, for 4K or 16K drams</td>
</tr>
<tr>
<td>74601</td>
<td>dynamic memory refresh controller, transparent and burst modes, for 64K drams</td>
</tr>
<tr>
<td>74602</td>
<td>dynamic memory refresh controller, cycle steal and burst modes, for 4K or 16K drams</td>
</tr>
<tr>
<td>74603</td>
<td>dynamic memory refresh controller, cycle steal and burst modes, for 64K drams</td>
</tr>
<tr>
<td>74604</td>
<td>octal 2-input multiplexer with latch, high-speed, with three-state outputs</td>
</tr>
<tr>
<td>74605</td>
<td>latch, high-speed, with open collector outputs</td>
</tr>
<tr>
<td>74606</td>
<td>octal 2-input multiplexer with latch, glitch-free, with three-state outputs</td>
</tr>
<tr>
<td>74607</td>
<td>octal 2-input multiplexer with latch, glitch-free, with open collector outputs</td>
</tr>
<tr>
<td>74608</td>
<td>memory cycle controller</td>
</tr>
<tr>
<td>74610</td>
<td>memory mapper, latched, three-state outputs</td>
</tr>
<tr>
<td>74611</td>
<td>memory mapper, latched, open collector outputs</td>
</tr>
<tr>
<td>74612</td>
<td>memory mapper, three-state outputs</td>
</tr>
<tr>
<td>74613</td>
<td>memory mapper, open collector outputs</td>
</tr>
<tr>
<td>74620</td>
<td>octal bus transceiver, inverting, three-state outputs</td>
</tr>
<tr>
<td>74621</td>
<td>octal bus transceiver, noninverting, open collector outputs</td>
</tr>
<tr>
<td>74622</td>
<td>octal bus transceiver, inverting, open collector outputs</td>
</tr>
<tr>
<td>74623</td>
<td>octal bus transceiver, noninverting, three-state outputs</td>
</tr>
<tr>
<td>74624</td>
<td>voltage-controlled oscillator with enable control, range control, two-phase outputs</td>
</tr>
</tbody>
</table>
74625  dual voltage-controlled oscillator with two-phase outputs
74626  dual voltage-controlled oscillator with enable control, two-phase outputs
74627  dual voltage-controlled oscillator
74628  voltage-controlled oscillator with enable control, range control, external temperature compensation, and two-phase outputs
74629  dual voltage-controlled oscillator with enable control, range control
74630  16-bit error detection and correction (EDAC) with three-state outputs
74631  16-bit error detection and correction with open collector outputs
74632  32-bit error detection and correction
74638  octal bus transceiver with inverting three-state outputs
74639  octal bus transceiver with noninverting three-state outputs
74640  octal bus transceiver with inverting three-state outputs
74641  octal bus transceiver with noninverting open collector outputs
74642  octal bus transceiver with inverting open collector outputs
74643  octal bus transceiver with mix of inverting and noninverting three-state outputs
74644  octal bus transceiver with mix of inverting and noninverting open collector outputs
74645  octal bus transceiver
74646  octal bus transceiver/latch/multiplexer with noninverting three-state outputs
74647  octal bus transceiver/latch/multiplexer with noninverting open collector outputs
74648  octal bus transceiver/latch/multiplexer with inverting three-state outputs
74649  octal bus transceiver/latch/multiplexer with inverting open collector outputs
74651  octal bus transceiver/register with inverting three-state outputs
74652  octal bus transceiver/register with noninverting three-state outputs
74653  octal bus transceiver/register with inverting three-state and open collector outputs
74654  octal bus transceiver/register with noninverting three-state and open collector outputs
74658  octal bus transceiver with Parity, inverting
74659  octal bus transceiver with Parity, noninverting
74664  octal bus transceiver with Parity, inverting
74665  octal bus transceiver with Parity, noninverting
74668  synchronous 4-bit decade Up/down counter
74669  synchronous 4-bit binary Up/down counter
74670  4 by 4 register File with three-state outputs
74671  4-bit bidirectional shift register/latch/multiplexer with three-state outputs
74672  4-bit bidirectional shift register/latch/multiplexer with three-state outputs
74673  16-bit serial-in serial-out shift register with output storage registers, three-state outputs
74674  16-bit parallel-in serial-out shift register with three-state outputs
74677  16-bit address comparator with enable
74678  16-bit address comparator with latch
74679  12-bit address comparator with latch
74680  12-bit address comparator with enable
74681  4-bit parallel binary accumulator
74682  8-bit magnitude comparator
74683  8-bit magnitude comparator with open collector outputs
74684  8-bit magnitude comparator
74685  8-bit magnitude comparator with open collector outputs
74686  8-bit magnitude comparator with enable
74687  8-bit magnitude comparator with enable
74688  8-bit equality comparator
74689  8-bit magnitude comparator with open collector outputs
74690  three state outputs
74691  4-bit binary counter/latch/multiplexer with asynchronous reset, three-state outputs
74692  4-bit decimal counter/latch/multiplexer with synchronous reset, three-state outputs
74693  4-bit binary counter/latch/multiplexer with synchronous reset, three-state outputs
74694  4-bit decimal counter/latch/multiplexer with synchronous and asynchronous resets, three-state outputs
74695  4-bit binary counter/latch/multiplexer with synchronous and asynchronous resets, three-state outputs
74696  4-bit decimal counter/register/multiplexer with asynchronous reset, three-state outputs
74697  4-bit binary counter/register/multiplexer with asynchronous reset, three-state outputs
74698  4-bit decimal counter/register/multiplexer with synchronous reset, three-state outputs
74699  4-bit binary counter/register/multiplexer with synchronous reset, three-state outputs
74716  programmable decade counter
74718  programmable binary counter
74724  voltage controlled multivibrator
74740  octal buffer/Line driver, inverting, three-state outputs
74741  octal buffer/Line driver, noninverting, three-state outputs, mixed enable polarity
74744  octal buffer/Line driver, noninverting, three-state outputs
74748  8 to 3-line priority encoder
74779  8-bit bidirectional binary counter (3-state)
74783  synchronous address multiplexer
74790  error detection and correction (EDAC)
74794  8-bit register with readback
74795  octal buffer with three-state outputs
74796  octal buffer with three-state outputs
74797  octal buffer with three-state outputs
74798  octal buffer with three-state outputs
74804  hex 2-input NAND drivers
74805  hex 2-input NOR drivers
74808  hex 2-input AND drivers
74832  hex 2-input OR drivers
74848  8 to 3-line priority encoder with three-state outputs
74873  octal transparent latch
74874  octal d-type flip-flop
74876  octal d-type flip-flop with inverting outputs
74878  dual 4-bit d-type flip-flop with synchronous clear, noninverting three-state outputs
74879  dual 4-bit d-type flip-flop with synchronous clear, inverting three-state outputs
74880  octal transparent latch with inverting outputs
74881  arithmetic logic unit
74882  32-bit lookahead carry generator
74888  8-bit slice processor
74901  hex inverting TTL buffer
74902  hex non-inverting TTL buffer
74903  hex inverting CMOS buffer
74904  hex non-inverting CMOS buffer
74905  12-Bit successive approximation register
74906  hex open drain n-channel buffers
74907  hex open drain p-channel buffers
74908  dual CMOS 30V relay driver
74909  quad voltage comparator
74910  256x1 CMOS static RAM
74911  4 digit expandable display controller
74912  6 digit BCD display controller and driver
74914  hex schmitt trigger with extended input voltage
74915  seven segment to BCD decoder
74917  6 digit Hex display controller and driver
74918  dual CMOS 30V relay driver
74920  256x4 CMOS static RAM
74921  256x4 CMOS static RAM
74922  16-key encoder
74923  20-key encoder
74925  4-digit counter/display driver
74926  4-digit counter/display driver
74927  4-digit counter/display driver
74928  4-digit counter/display driver
74929  1024x1 CMOS static RAM
74930  1024x1 CMOS static RAM
74932  phase comparator
74933  address bus comparator
74934  =ADC0829 ADC, see corresponding NSC datasheet
74935  3.5-digit digital voltmeter (DVM) support chip for multiplexed 7-segment displays
74936  3.75-digit digital voltmeter (DVM) support chip for multiplexed 7-segment displays
74937  =ADC3511 ADC, see corresponding NSC datasheet
74938  =ADC3711 ADC, see corresponding NSC datasheet
74941  octal bus/line drivers/line receivers
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>74945</td>
<td>4 digit up/down counter with decoder and driver</td>
</tr>
<tr>
<td>74947</td>
<td>4 digit up/down counter with decoder and driver</td>
</tr>
<tr>
<td>74948</td>
<td>=ADC0816 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74949</td>
<td>=ADC0808 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74949</td>
<td>=ADC0808 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>741005</td>
<td>hex inverting buffer with open-collector output</td>
</tr>
<tr>
<td>741035</td>
<td>hex noninverting buffers with open-collector outputs</td>
</tr>
<tr>
<td>742960</td>
<td>error detection and correction (EDAC)</td>
</tr>
<tr>
<td>742961</td>
<td>edac bus buffer, inverting</td>
</tr>
<tr>
<td>742962</td>
<td>edac bus buffer, noninverting</td>
</tr>
<tr>
<td>742968</td>
<td>dynamic memory controller</td>
</tr>
<tr>
<td>742969</td>
<td>memory timing controller for use with EDAC</td>
</tr>
<tr>
<td>742970</td>
<td>memory timing controller for use without EDAC</td>
</tr>
<tr>
<td>744002</td>
<td>single 3 input OR-AND Gate;</td>
</tr>
<tr>
<td>744017</td>
<td>dual 4-bit shift registers</td>
</tr>
<tr>
<td>744020</td>
<td>5-stage ÷10 Johnson counter</td>
</tr>
<tr>
<td>744024</td>
<td>14-stage binary counter</td>
</tr>
<tr>
<td>744032</td>
<td>6-stage ripple carry binary counter</td>
</tr>
<tr>
<td>744058</td>
<td>BCD to decimal decoder</td>
</tr>
<tr>
<td>744059</td>
<td>phase-locked loop decoder and voltage-controlled oscillator</td>
</tr>
<tr>
<td>744066</td>
<td>hex inverting buffer</td>
</tr>
<tr>
<td>744067</td>
<td>hex buffer/converter (non-inverting)</td>
</tr>
<tr>
<td>744068</td>
<td>high-speed CMOS 8-channel analog multiplexer/demultiplexer</td>
</tr>
<tr>
<td>744070</td>
<td>dual 4-channel analog multiplexer/demultiplexers</td>
</tr>
<tr>
<td>744075</td>
<td>triple 2-channel analog multiplexer/demultiplexers</td>
</tr>
<tr>
<td>744076</td>
<td>programmable divide-by-N counter</td>
</tr>
<tr>
<td>744077</td>
<td>14-stage binary ripple counter with oscillator</td>
</tr>
<tr>
<td>744084</td>
<td>7 stage ripple carry binary counter</td>
</tr>
<tr>
<td>744094</td>
<td>BCD to 7-segment decoder</td>
</tr>
<tr>
<td>744102</td>
<td>8-bit three-state shift register/latch</td>
</tr>
<tr>
<td>744107</td>
<td>quad analog switch</td>
</tr>
<tr>
<td>744111</td>
<td>dual 4-bit synchronous binary counter</td>
</tr>
<tr>
<td>744130</td>
<td>dual retriggerable precision monostable multivibrator</td>
</tr>
</tbody>
</table>
747007  hex buffer
747266  quad 2-input XNOR gate
7429841 10-bit bus-interface D-type latch with 3-state outputs
7440103  presettable 8-bit synchronous down counter
7440105 4-bit by 16-word FIFO register.

**Short Answer Type Questions**

1. Define AND logic gate.
2. Define OR Logic gate.
3. Define NOT logic gate.
4. Draw the logic symbols of AND, OR and NOT gates.
5. Draw the logic symbols of NAND, NOR gates.
6. Define EX-OR gate.
8. Mention the Digital logic families.
9. Write any two IC numbers of Logic gates.

**Long Answer Type Questions**

1. Draw and explain the working of AND, OR gates with truth table.
2. Explain working of universal NAND gate.
3. Explain working of universal NOR gate.
4. Explain working of an EX-OR gate with truth table.
5. Explain Fan-In, Fan-Out, Propagation delay, Power dissipation, Threshold voltage and Speed.

**On Job Training**

1. Study the basic logic gates AND, OR, NOT, EX-OR, NAND, NOR gates with Truth tables.
2. Study the digital IC families with their characteristics.
LEARNING OBJECTIVES

• To study combinational logic circuit.
• To study half adder by using EX-OR gate and AND gate
• To study half adder circuit by using NAND, NOR gates.
• To study 4-bit parallel full adders
• To study the multiplexer working
• To study the operation of 1-4 De-multiplexor.
• To study 3x8 Decoders
• To study the applications of multiplexors and De-multiplexors.
• To study the popular digital IC’s and their applications.

3.0 INTRODUCTION

Combinational Logic Circuits are made up from basic logic NAND, NOR or NOT gates that are “combined” or connected together to produce more complicated switching circuits. These logic gates are the building blocks of combinational logic circuits. An example of a combinational circuit is a decoder, which converts the binary code data present at its input into a number of different output lines, one at a time producing an equivalent decimal code at its output.
Combinational logic circuits can be very simple or very complicated and any combinational circuit can be implemented with only NAND and NOR gates as these are classed as “universal” gates. The three main ways of specifying the function of a combinational logic circuit are:

- **Truth Table** Truth tables provide a concise list that shows the output values in tabular form for each possible combination of input variables.

- **Boolean Algebra** Forms an output expression for each input variable that represents a logic “1”

- **Logic Diagram** Shows the wiring and connections of each individual logic gate that implements the circuit.

### 3.1 Combinational Logic Circuit

![Combinational Logic Circuit Diagram](Fig. 3.1 Combinational Logic Circuit)

One of the most common uses of combinational logic is in Multiplexer and De-multiplexer type circuits. Here, multiple inputs or outputs are connected to a common signal line and logic gates are used to decode an address to select a single data input or output switch. A multiplexer consist of two separate components, a logic decoder and some solid state switches, but before we can discuss multiplexers, decoders and de-multiplexers in more detail we first need to understand how these devices use these “solid state switches” in their design.
Solid State Switches

Standard TTL logic devices made up from Transistors can only pass signal currents in one direction only making them “uni-directional” devices and poor imitations of conventional electro-mechanical switches or relays. However, some CMOS switching devices made up from FET’s act as near perfect “bi-directional” switches making them ideal for use as solid state switches.

Solid state switches come in a variety of different types and ratings, and there are many different applications for using solid state switches. They can basically be sub-divided into 3 different main groups for switching applications and in this combinational logic section we will only look at the Analogue type of switch but the principal is the same for all types including digital.

Solid State Switch Applications

- Analogue Switches Data & Process Control, Video & Audio Switching, Instrumentation ...etc.
- Digital Switches High Speed Data Transmission, Switching & Routing, LAN’s, USB ...etc.
- Power Switches Power Supplies and general “Standby Power” Switching Applications ...etc.

Analogue Bilateral Switches

- Analogue or “Analog” switches are those types that are used to switch data or signal currents when they are in their “ON” state and block them when they are in their “OFF” state. The rapid switching between the “ON” and the “OFF” state is usually controlled by a digital signal applied to the control gate of the switch. An ideal analogue switch has zero resistance when “ON” (or closed), and infinite resistance when “OFF” (or open) and switches with $R_{ON}$ values of less than $1\Omega$ are commonly available.

Solid State Analog Switch

By connecting an N-channel MOSFET in parallel with a P-channel MOSFET allows signals to pass in either direction making it a Bi-directional switch and as to whether the N-channel or the P-channel device carries more signal current will depend upon the ratio between the input to the output voltage. The two MOSFETs are switched “ON” or “OFF” by two internal non-inverting and inverting amplifiers.
Contact Types

Just like mechanical switches, analogue switches come in a variety of forms or contact types, depending on the number of “poles” and “throws” they offer. Thus, terms such as “SPST” (single-pole single throw) and “SPDT” (single-pole double-throw) also apply to solid state analogue switches with “make-before-break” and “break-before-make” configurations available.

Analogue Switch Types

Individual analogue switches can be grouped together into standard IC packages to form devices with multiple switching configurations of SPST and SPDT as well as multi channel multiplexers. The most common and simplest analogue switch IC is the 74HC4066 which has 4 independent bi-directional
“ON/OFF” Switches within a single package but the most widely used variants of the CMOS analogue switch are those described as “Multi-way Bilateral Switches” otherwise known as the “Multiplexer” and “De-multiplexer” IC’s.

3.2 Half Adder Circuit using EX-Or Gate and AND Gate

Half Adder is a combinational circuit that performs addition of two bits. It has two inputs and two outputs. The two I/Ps are the two 1-bit numbers A and B designated as augend and addend bits. The two O/Ps are the sum ‘S’ of A and B and the carry bit, denoted by ‘C’.

**Truth Table** of a half adder can be derived by performing binary addition of augend and addend bits as follows:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Sum</th>
<th>Carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 3.4 Truth table of Half Adder

From the truth table, Boolean Expression can be derived as:

\[ S = A'B + AB' = A \oplus B \]

\[ C = AB \]

A Half Adder circuit can be implemented using XOR & AND logic gates. These implementations are shown in the image below:

Fig. 3.5 Half Adder
3.3 Half Adder Using NAND and NOR Gates

Half adder is a combinational arithmetic circuit that adds two numbers and produces a sum bit (S) and carry bit (C) as the output. If A and B are the input bits, then sum bit (S) is the X-OR of A and B and the carry bit (C) will be the NAND of A and B. From this it is clear that a half adder circuit can be easily constructed using one NOR gate and one NAND gate. Half adder is the simplest of all adder circuit, but it has a major disadvantage. The half adder can add only two input bits (A and B) and has nothing to do with the carry if there is any in the input. So if the input to a half adder have a carry, then it will be neglected it and adds only the A and B bits. That means the binary addition process is not complete and that’s why it is called a half adder. NAND gates or NOR gates can be used for realizing the half adder in universal logic and the relevant circuit diagrams are shown in the figure below.

![Fig. 3.6 Half Adder using NAND Gate](image1.png)  
![Fig. 3.7 Using NOR Gate](image2.png)

3.4 Full Adder Circuit

The full-adder circuit adds three one-bit binary numbers (C A B) and outputs two one-bit binary numbers, a sum (S) and a carry (C1). The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. binary numbers. The carry input for the full-adder circuit is from the carry output from the circuit “above” itself in the cascade. The carry output from the full adder is fed to another full adder “below” itself in the cascade.
**3.5 Two Half Adders and an OR-Gate Constitute Full-Adder**

Boolean expression above can be implemented with a two-input EX-OR gate provided that one of the inputs is Cin and the other input is the output of another two-input EX-OR gate with A and B as its inputs. Similarly, Boolean expression above can be implemented by ORing two minterms. One of them is the AND output of A and B. The other is also the output of an AND gate whose
inputs are Cin and the output of an EX-OR operation on A and B. The whole idea of writing the Boolean expressions in this modified form was to demonstrate the use of a half-adder circuit in building a full adder. Figure 3.7(a) shows logic implementation of Equations above. The full adder of the type described above forms the basic building block of binary adders. However, a single full adder circuit can be used to add one-bit binary numbers only. A cascade arrangement of these adders can be used to construct adders capable of adding binary numbers with a larger number of bits. For example, a four-bit binary adder would require four full adders of the type to be connected in cascade. Figure 3.10 shows such an arrangement. \((A_3, A_2, A_1, A_0)\) and \((B_3, B_2, B_1, B_0)\) are the two binary numbers to be added, with \(A_0\) and \(B_0\) representing LSBs and \(A_3\) and \(B_3\) representing MSBs of the two numbers.

![Fig. 3.10 Two Half Adders and OR Gate Full Adder](image)

### 3.6 Working of 4 Bit Parallel Adder using Full-Adders

The ripple carry adder is constructed by cascading full adders (FA) blocks in series. One full adder is responsible for the addition of two binary digits at any stage of the ripple carry. The carryout of one stage is fed directly to the carry-in of the next stage.

A number of full adders may be added to the ripple carry adder or ripple carry adders of different sizes may be cascaded in order to accommodate binary vector strings of larger sizes. For an \(n\)-bit parallel adder, it requires \(n\) computational elements (FA). Figure 3.11 shows an example of a parallel adder: a 4-bit ripple-carry adder. It is composed of four full adders. The augend s bits of \(x\) are added to the addend bits of \(y\) respectfully of their binary position. Addition creates a sum and a carry out. The carry out is then transmitted to the carry-in of the next higher-order bit. The final result creates a sum of four bits plus a carry out (c4). Even though this is a simple adder and can be used to add unrestricted bit length numbers, it is however not very efficient when large bit numbers are used.
3.7 Working of 4 x 1 Multiplexer

Multiplexer (MUX), sometimes called Data Selector is a combinational logic circuit that selects one of \(2^n\) inputs on the and route it to the output. It has \(2^n\) inputs, \(n\) select lines (selectors) that identify which input will be provided to the output, and only one output. Fig. 3.12 shows the block diagram and the truth table for a 4 x 1 MUX.

![4 x 1 Multiplexer](image)

<table>
<thead>
<tr>
<th>Strobe</th>
<th>(S_1)</th>
<th>(S_0)</th>
<th>(O)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(X)</td>
<td>(X)</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>(I_0)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>(I_1)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>(I_2)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>(I_3)</td>
</tr>
</tbody>
</table>

Large multiplexer can be implemented using smaller size multiplexers. For example, consider a 8 x 1 MUX, this multiplexer can be implemented using two 4 x 1 MUX’s one 2 x 1 MUX.

3.8 Working of 1 to 4 Demultiplexer

The Demultiplexer is combinational logic circuit that performs the reverse operation of Multiplexer. It has only one input, \(n\) selectors and \(2n\) output. Depending on the combination of the select lines, one of the output will be
selected to take the state of input. Fig. 3.14 shows the block diagram and truth table for 1 x 4 demultiplexer. By applying logic ‘1’ to the input, the circuit will do the same function of the typical 2 to 4 decoder.

![1 x 4 Demultiplexer Diagram]

**Fig. 3.14 1-4 Demultiplexer**

<table>
<thead>
<tr>
<th>( S_1 )</th>
<th>( S_0 )</th>
<th>( O_0 )</th>
<th>( O_1 )</th>
<th>( O_2 )</th>
<th>( O_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Fig. 3.15 Truth table Demultiplexer**

### 3.9 3 x 8 Decoder Working

In a three to eight decoder, there are three inputs and eight outputs, as shown in figure 3.16. \( A_0 \) is the least significant variable, while \( A_2 \) is the most significant variable. The three inputs are decoded into eight outputs. That is, binary values at the input form a combination, and based on this combination, the corresponding output line is activated. Each output represents one minterm.

- For example, for input combination \( A_2A_1A_0 = 001 \), output line \( D1 \) equals 1 while all other output lines equal 0’s.

- It should be noted that at any given instance of time, one and only one output line can be activated. It is also obvious from the fact that only one
combination is possible at the input at a time, so the corresponding output line is activated.

Fig. 3.16  A 3-8 Decoder with enable

![3-to-8 Decoder Diagram]

<table>
<thead>
<tr>
<th>Dec. Code</th>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A₂ A₁ A₀</td>
<td>D₀ D₁ D₂ D₃ D₄ D₅ D₆ D₇</td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>1 0 0 0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>1 0 0</td>
<td>0 1 0 0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>2 0 1</td>
<td>0 0 1 0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>3 0 1</td>
<td>0 0 0 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>4 1 0</td>
<td>0 0 0 0 1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>5 1 0</td>
<td>0 0 0 0 0 1 0 0</td>
<td></td>
</tr>
<tr>
<td>6 1 1</td>
<td>0 0 0 0 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>7 1 1</td>
<td>0 0 0 0 0 0 0 1</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 3.17  3-8 Decoder Truth Table

Since each input combination represents one minterm, the truth table (table ) contains eight output functions, from D₀ to D₇, seven, where each function represents one and only one minterm. Thus function D₀ is $\overline{A₂} \overline{A₁} \overline{A₀}$. Similarly function D₇ is $A₂ A₁ A₀$. The corresponding circuit is given in . In this figure 3.18 , the three inverters provide complement of the inputs, and each one of the AND gates generates one of the minterms. It is also possible to add an Enable input to this decoder.
3.10 Applications of Multiplexers and Decoders

Applications of Multiplexers

Multiplexers switch two or more input signals to an output port. This allows multiple information streams to be sent over one transmission channel. Since the multiplexer spends some time on each input, the time is divided between the inputs.

Applications are cellphone systems, instrumentation, and any other function where only one transmission channel (e.g. a radio transmitter) is available, yet information from several sources must be sent.

Applications of Decoders

Decoders are used to analyze data streams for a certain data code and give an output if the data is present like an address to a peripheral unit that needs service.

3.11 Popular Digital IC’s and Applications

The following is a list of 7400 series digital logic integrated circuits. The SN7400 series originated with TTL integrated circuits made by Texas
Instruments. Because of the popularity of these parts, they were second-sourced by other manufacturers who kept the 7400 sequence number as an aid to identification of compatible parts. As well, compatible TTL parts originated by other manufacturers were second sourced in the TI product line under a 74xxx series part number.

- 7418: Dual 4-input NAND gate with Schmitt trigger inputs
- 7419: Hex Schmitt trigger inverter
- 7420: Dual 4-input NAND gate
- 7421: Dual 4-input AND gate
- 7422: Dual 4-input NAND gate with open collector outputs
- 7423: Expandable dual 4-input NOR gate with strobe
- 7424: Quad 2-input NAND gate gates with schmitt-trigger line-receiver inputs.
- 7425: Dual 4-input NOR gate with strobe
- 7426: Quad 2-input NAND gate with 15 v open collector outputs
- 7427: Triple 3-input NOR gate
- 741G27: Single 3-input NOR gate
- 7428: Quad 2-input NOR buffer
- 7430: 8-input NAND gate
- 7431: Hex delay elements
- 7432: Quad 2-input OR gate
- 741G32: Single 2-input OR gate
- 7433: Quad 2-input NOR buffer with open collector outputs
- 7436: Quad 2-input NOR gate (different pinout than 7402)
- 7437: Quad 2-input NAND buffer
- 7438: Quad 2-input NAND buffer with open collector outputs
- 7439: Quad 2-input NAND buffer
- 7440: Dual 4-input NAND buffer
- 7441: BCD to decimal decoder/Nixie tube driver
- 7442: BCD to decimal decoder
- 7443: Excess-3 to decimal decoder
- 7444: Excess-3-Gray code to decimal decoder
- 7445: BCD to decimal decoder/driver
- 7446: BCD to seven-segment display decoder/driver with 30 v open collector outputs
- 7447: BCD to 7-segment decoder/driver with 15 v open collector outputs
- 7448: BCD to 7-segment decoder/driver with Internal Pullups
7449  BCD to 7-segment decoder/driver with open collector outputs
7450  dual 2-wide 2-input AND-OR-invert gate (one gate expandable)
7451  dual 2-wide 2-input AND-OR-invert gate
7452  expandable 4-wide 2-input AND-OR gate
7453  expandable 4-wide 2-input AND-OR-invert gate
7454  4-wide 2-input AND-OR-invert gate
7455  2-wide 4-input AND-OR-invert Gate (74H version is expandable)
7456  50:1 frequency divider
7457  60:1 frequency divider
7458  2-input & 3-input AND-OR Gate
7459  2-input & 3-input AND-OR-invert Gate
7460  dual 4-input expander
7461  triple 3-input expander
7462  3-2-2-3-input AND-OR expander
7463  hex current sensing interface gates
7464  4-2-3-2-input AND-OR-invert gate
7465  4-2-3-2 input AND-OR-invert gate with open collector output
7468  dual 4 bit decade counters
7469  dual 4 bit binary counters
7470  AND-gated positive edge triggered J-K flip-flop with preset and clear
74H71  AND-or-gated J-K master-slave flip-flop with preset
74L71  AND-gated R-S master-slave flip-flop with preset and clear
7472  AND gated J-K master-slave flip-flop with preset and clear
7473  dual J-K flip-flop with clear
7474  dual D positive edge triggered flip-flop with preset and clear
7475  4-bit bistable latch
7476  dual J-K flip-flop with preset and clear
7477  4-bit bistable latch
74H78  dual positive pulse triggered J-K flip-flop with preset, common clock, and common clear (different pinout than 74L78 / 74Ls78)
74L78  dual positive pulse triggered J-K flip-flop with preset, common clock, and common clear
74LS74 dual negative edge triggered J-K flip-flop with preset, common clock, and common clear
7479 dual D flip-flop
741G79 single D-type flip-flop positive edge trigger non-inverting output
7480 gated full adder
741G80 single D-type flip-flop positive edge trigger inverting output
7481 16-bit random access memory
7482 2-bit binary full adder
7483 4-bit binary full adder
7484 16-bit random access memory
7485 4-bit magnitude comparator
7486 quad 2-input XOR gate
741G86 single 2 input exclusive-OR gate
7487 4-bit true/complement/zero/one element
7488 256-bit read-only memory
7489 64-bit random access memory
7490 decade counter (separate divide-by-2 and divide-by-5 sections)
7491 8-bit shift register, serial In, serial out, gated input
7492 divide-by-12 counter (separate divide-by-2 and divide-by-6 sections)
7493 4-bit binary counter (separate divide-by-2 and divide-by-8 sections)
7494 4-bit shift register, dual asynchronous presets
7495 4-bit shift register, parallel In, parallel out, serial input
7496 5-bit parallel-In/parallel-out shift register, asynchronous preset
7497 synchronous 6-bit binary rate multiplier
741G97 configurable multiple-function gate
7498 4-bit data selector/storage register
7499 4-bit bidirectional universal shift register
74100 dual 4-bit bistable latch
74101 AND-or-gated J-K negative-edge-triggered flip-flop with preset
74102 AND-gated J-K negative-edge-triggered flip-flop with preset and clear
74103 dual J-K negative-edge-triggered flip-flop with clear
74104 J-K master-slave flip-flop
Short Answer Type Questions

1. What is Combinational logic circuit.
2. What are the functions of Half Adder
3. Draw the Full Adder circuit.
4. What are the applications of full adder.
5. What are the applications of four bit parallel adder.
6. Write applications of Multiplexer.
7. What are the applications of Decoders.
8. Write any two Digital IC numbers.
9. Write applications of Digital IC’s.

Long Answer Type Questions

1. Draw and explain Half adder circuit with truth table.
2. Draw Half adder and explain the working by using NAND Gates.
3. Explain working of Half Adder by using NOR Gates.
4. Draw and Explain Full adder circuit with truth table.
5. Draw and Explain Four bit parallel adder by using full adder.
6. Draw and explain 4 x 1 multiplexer.
7. Draw and explain working of 1 x 4 Demultiplexer.
8. Draw and explain working of 3 x 8 Decoder.

On Job Training / Practical Questions

1. Study the half adder circuit by using AND and X-OR gate.
2. Study the half adder circuit by using NAND, NOR gates.
3. Study the full adder circuit with truth table.
4. Study the full adder circuit by using two half adders and an OR gate.
5. Study the operation of 4 x 1 multiplexer.
6. Study the operation of 1 x 4 demultiplexer.
7. Study the working of 3 x 8 decoder.
Learning Objectives

After studying this unit, the student will be able to know

- Definition of sequential logic circuit
- Study of NAND and NOR latches with truth tables.
- Study of clock pulse triggering
- Study of SR flip flop by using NAND gates
- Study of preset and clear inputs
- Study of JK, SR flip flop working with truth table and applications
- Study of clocked D, T flip flops working with truth tables and applications
- Study of shift register and applications
- Study of 4 bit shift left and shift right registers by using IC(7475)
- Study of parallel in and parallel out shift register working
- Study of universal shift register by using IC(74194)
· Study of Asynchronous and asynchronous/decade counters and counters and applications

· Study Applications of flip flop registers counters and their IC numbers

4.0 Introduction to Sequential Circuits

Digital electronics is classified into combinational logic and sequential logic. Combinational logic output depends on the inputs levels, whereas sequential logic output depends on stored levels and also the input levels.

The memory elements are devices capable of storing binary info. The binary info stored in the memory elements at any given time defines the state of the sequential circuit. The input and the present state of the memory element determines the output. Memory elements next state is also a function of external inputs and present state. A sequential circuit is specified by a time sequence of inputs, outputs, and internal states.

4.1 Sequential Logic Circuit

There are two types of sequential circuits. Their classification depends on the timing of their signals:

- Synchronous sequential circuits
- Asynchronous sequential circuits

Asynchronous Sequential Circuit

This is a system whose outputs depend upon the order in which its input variables change and can be affected at any instant of time. Gate-type asynchronous systems are basically combinational circuits with feedback paths.
Because of the feedback among logic gates, the system may, at times, become unstable. Consequently they are not often used.

**Synchronous Sequential Circuits**

This type of system uses storage elements called flip-flops that are employed to change their binary value only at discrete instants of time. Synchronous sequential circuits use logic gates and flip-flop storage devices. Sequential circuits have a clock signal as one of their inputs. All state transitions in such circuits occur only when the clock value is either 0 or 1 or happen at the rising or falling edges of the clock depending on the type of memory elements used in the circuit. Synchronization is achieved by a timing device called a clock pulse generator. Clock pulses are distributed throughout the system in such a way that the flip-flops are affected only with the arrival of the synchronization pulse. Synchronous sequential circuits that use clock pulses in the inputs are called clocked-sequential circuits. They are stable and their timing can easily be broken down into independent discrete steps, each of which is considered separately.
A clock signal is a periodic square wave that indefinitely switches from 0 to 1 and from 1 to 0 at fixed intervals. Clock cycle time or clock period: the time interval between two consecutive rising or falling edges of the clock.

### 4.2 Working of NAND and NOR Latches

#### NAND based (S - R) Latch

The S - R latch can be constructed using two NAND gates connected back to back as shown.

The truth table for NAND based S - R latch is shown:

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q(_n+2)</th>
<th>Q(_n+2)</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>?</td>
<td>?</td>
<td>Forbidden</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Set</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Q(_n)</td>
<td>Q(_n)</td>
<td>No Change</td>
</tr>
</tbody>
</table>

#### NAND based (S R) Latch:

The S - R latch can be constructed using two NAND gates connected back to back as shown.

The truth table for NAND based S - R latch is shown:

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q(_n+2)</th>
<th>Q(_n+2)</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>?</td>
<td>?</td>
<td>Forbidden</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Set</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Q(_n)</td>
<td>Q(_n)</td>
<td>No Change</td>
</tr>
</tbody>
</table>

**State Diagram and Characteristic Equation Of SR latch:**

The S - R latch can be constructed using two NAND gates connected back to back as shown.

Truth Table for Nand Based SR Latch
4.3 Level Triggering and Edge Triggering by using clock

Level triggering means the specified action occurs based on the steady state value of the input. An SR flip flop is an example of this - so long as S is true, the flip flop will stay set.

Edge triggering, on the other hand, means the specified action occurs on the edge of the transition of the input, either from low to high or from high to low. The clock input of a D flip flop is an example of this - once the event occurs, the input must go back to its prior state and then generate the triggering edge again.

4.4 SR Flip Flop - Working by using NAND Gates

The SR flip-flop, also known as a SR Latch, can be considered as one of the most basic sequential logic circuit possible. This simple flip-flop is basically a one-bit memory bistable device that has two inputs, one which will “SET” the device (meaning the output = “1”), and is labelled S and another which will “RESET” the device (meaning the output = “0”), labelled R. Then the SR description stands for “Set-Reset”. The reset input resets the flip-flop back to its original state with an output Q that will be either at a logic level “1” or logic “0” depending upon this set/reset condition.

A basic NAND gate SR flip-flop circuit provides feedback from both of its outputs back to its opposing inputs and is commonly used in memory circuits to store a single data bit. Then the SR flip-flop actually has three inputs, Set, Reset and its current output Q relating to its current state or history. The term “Flip-flop” relates to the actual operation of the device, as it can be “flipped” into one logic Set state or “flopped” back into the opposing logic Reset state.

The NAND Gate SR Flip-Flop

The simplest way to make any basic single bit set-reset SR flip-flop is to connect together a pair of cross-coupled 2-input NAND gates as shown, to form a Set-Reset Bistable also known as an active LOW SR NAND Gate Latch, so that there is feedback from each output to one of the other NAND
gate inputs. This device consists of two inputs, one called the Set, S and the other called the Reset, R with two corresponding outputs Q and its inverse or complement Q (not-Q) as shown below.

![Fig. 4.5 S-R Flip Flop using NAND Gates with Clock pulse](image)

**The Set State**

Consider the circuit shown above. If the input R is at logic level “0” (R = 0) and input S is at logic level “1” (S = 1), the NAND gate Y has at least one of its inputs at logic “0” therefore, its output Q must be at a logic level “1” (NAND Gate principles). Output Q is also fed back to input “A” and so both inputs to NAND gate X are at logic level “1”, and therefore its output Q must be at logic level “0”. Again NAND gate principals. If the reset input R changes state, and goes HIGH to logic “1” with S remaining HIGH also at logic level “1”, NAND gate Y inputs are now R = “1” and B = “0”. Since one of its inputs is still at logic level “0” the output at Q still remains HIGH at logic level “1” and there is no change of state. Therefore, the flip-flop circuit is said to be “Latched” or “Set” with Q = “1” and Q = “0”.

**Reset State**

In this second stable state, Q is at logic level “0”, (not Q = “0”) its inverse output at Q is at logic level “1”, (Q = “1”), and is given by R = “1” and S = “0”. As gate X has one of its inputs at logic “0” its output Q must equal logic level “1” (again NAND gate principles). Output Q is fed back to input “B”, so both inputs to NAND gate Y are at logic “1”, therefore, Q = “0”. If the set input, S now changes state to logic “1” with input R remaining at logic “1”, output Q still remains LOW at logic level “0” and there is no change of state. Therefore, the flip-flop circuits “Reset” state has also been latched and we can define this “set/reset” action in the following truth table.
Truth Table for this Set-Reset Function

<table>
<thead>
<tr>
<th>State</th>
<th>S</th>
<th>R</th>
<th>Q</th>
<th>Q</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Set Q &gt;&gt; 1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>No change</td>
</tr>
<tr>
<td>Reset</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Reset Q &gt;&gt; 0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>no change</td>
</tr>
<tr>
<td>Invalid</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>memory with Q=0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>memory with Q=1</td>
</tr>
</tbody>
</table>

It can be seen that when both inputs S = “1” and R = “1” the outputs Q and \( \overline{Q} \) can be at either logic level “1” or “0”, depending upon the state of inputs S or R BEFORE this input condition existed. However, input state R = “0” and S = “0” is an undesirable or invalid condition and must be avoided because this will give both outputs Q and \( \overline{Q} \) to be at logic level “1” at the same time and we would normally want Q to be the inverse of \( \overline{Q} \). However, if the two inputs are now switched HIGH again after this condition to logic “1”, both the outputs will go LOW resulting in the flip-flop becoming unstable and switch to an unknown data state based upon the unbalance. This unbalance can cause one of the outputs to switch faster than the other resulting in the flip-flop switching to one state or the other which may not be the required state and data corruption will exist. This unstable condition is known as its **Meta-stable** state.

Then, a bistable SR flip-flop or SR latch is activated or set by a logic “1” applied to its S input and deactivated or reset by a logic “1” applied to its R. The SR flip-flop is said to be in an “invalid” condition (Meta-stable) if both the set and reset inputs are activated simultaneously.

### 4.5 Need for Preset and Clear Inputs

The normal data inputs to a flip flop (D, S and R, or J and K) are referred to as synchronous inputs because they have effect on the outputs (Q and not-Q) only in step, or in sync, with the clock signal transitions. These extra inputs that I now bring to your attention are called asynchronous because they can set or reset the flip-flop regardless of the status of the clock signal. Typically, they’re called preset and clear:
When the preset input is activated, the flip-flop will be set \((Q=1, \text{not-}Q=0)\) regardless of any of the synchronous inputs or the clock. When the clear input is activated, the flip-flop will be reset \((Q=0, \text{not-}Q=1)\), regardless of any of the synchronous inputs or the clock. So, what happens if both preset and clear inputs are activated? Surprise, surprise: we get an invalid state on the output, where \(Q\) and not-\(Q\) go to the same state, the same as our old friend, the S-R latch! Preset and clear inputs find use when multiple flip-flops are ganged together to perform a function on a multi-bit binary word, and a single line is needed to set or reset them all at once.

Asynchronous inputs, just like synchronous inputs, can be engineered to be active-high or active-low. If they’re active-low, there will be an inverting bubble at that input lead on the block symbol, just like the negative edge-trigger clock inputs.

Sometimes the designations “PRE” and “CLR” will be shown with inversion bars above them, to further denote the negative logic of these inputs:

### 4.6 Clocked JK Flipflop using S-R Flipflop - Working

A JK flip-flop is a refinement of the SR flip-flop in that the indeterminate state of the SR type is defined in the JK type. Inputs J and K behave like inputs S and R to set and clear the flip-flop (note that in a JK flip-flop, the letter J is for set and the letter K is for clear). When logic 1 inputs are applied to both J and K simultaneously, the flip-flop switches to its complement state, i.e., if \(Q = 1\), it switches to \(Q = 0\) and vice versa.

A clocked JK flip-flop is shown in Figure 4. Output Q is ANDed with K and CP inputs so that the flip-flop is cleared during a clock pulse only if Q was
previously 1. Similarly, output Q’ is ANDed with J and CP inputs so that the flip-flop is set with a clock pulse only if Q’ was previously 1.

Note that because of the feedback connection in the JK flip-flop, a CP signal which remains a 1 (while J=K=1) after the outputs have been complemented once will cause repeated and continuous transitions of the outputs. To avoid this, the clock pulses must have a time duration less than the propagation delay through the flip-flop.

![JK Clocked Flip flop](image)

**Fig. 4.7 JK Clocked Flip flop**

<table>
<thead>
<tr>
<th>Q</th>
<th>J</th>
<th>K</th>
<th>(Q (t+1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Truth table for Jk Flip Flop
4.7 Level Clocked D and T Flip Flops - Working with Truth Table

The basic building blocks of combinational logic circuits are gates. In particular, AND, OR, and NOT gates (however, there are also, XOR, NAND, NOR, XNOR gates too).

The basic building blocks of sequential logic circuits are flip flops. Flip flops are devices that use a clock. Each flip flop can store one bit.

Here’s how a flip flop looks:

Fig. 4.8 Clock Pulse

Fig. 4.9 D & T Flipflops
Basically, a flip flop has two inputs. One input is a control input. For a D flip flop, the control input is labelled D. For a T flip flop, the control input is labelled T.

The other input is the clock. You can read about clock from the class notes on clock.

The clock input is usually drawn with a triangular input. These flip-flops are positive edge-triggered flip flops. This means that the flip flops can only change output values when the clock is at a positive edge. There are also negative edge triggered flip flops, which change on a negative edge, and level-triggered flip flops, that change only when the value is 1. We consider only positive edge-triggered flip flops.

When the clock is not at a positive edge, then the output value is held. That is, it does not change.

4.8 Truth tables of Edge triggered D & T Flipflops

A flip flop also has two outputs, Q and Q'. The output is really the bit that’s stored. Thus, the flip flop is always outputting the one bit of information.

But you might wonder “Doesn’t it have two bits of information? Q and Q’?” If you have two bits, you have four possible values. However, Q’ is the negation of Q which means you only have two possible outputs: Q = 0, Q’ = 1, Q = 1, Q’ = 0. Since the second output is always negated from the first, you don’t get any additional storage.

You might wonder why flip flops have two outputs, Q and Q’. It turns out you can design flip flops with NOR gates or NAND gates, with cycles (which is not allowed for combinational circuits). The design gives you Q’ basically for free, so that’s why flip flops have both the regular output and the negated output.

Although I rarely draw it, flip flops often have one additional input called asynchronous clear. It’s drawn at the top of the flip flop. This is an active low asynchronous clear. “Asynchronous” means “without a clock” (usually). Active low, means you need to set the value to 0, to make it active.

Thus, if you set the asynchronous clear to 0, it causes Q to be automatically set to 0. It does this, even if the clock has not reached a positive edge. That is, it sets Q to zero as fast as it can. The asynchronous clear is often used to reset flip flops to some initial value. Often, you see active low inputs because it consumes less power.

The flip flops have additional inputs. Flip flops and logic gates are powered
devices. They need an input for ground and usually 5 volts (although there are low voltage versions of the flip flops). However, they’re not drawn because they don’t affect how the flip flop behaves.

**Characteristic Tables**

The behavior of a flip flops can be described by a characteristic table which is basically a truth table.

**D Flip Flop Characteristic Table**

Here’s the characteristic table for a D flip flop.

<table>
<thead>
<tr>
<th>D</th>
<th>Q</th>
<th>Q⁺</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Set</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Set</td>
</tr>
</tbody>
</table>

*Fig. 4.10 Truth table of D Flip flop*

The D flip flop characteristic table has 3 columns. The first column is the value of D, a control input. The second column is the current state, that is the current value being output by Q. The third column is the next state, that is, the value of Q at the next positive edge. It’s labelled with Q and the superscript, + (the plus sign).

Sometimes, the current state is written as Q(t) which means the value of Q at the current time, t, and the next state is written as Q(t + 1) which means the value of Q at the next clock edge. However, I’ll usually write it as Q⁺.

The characteristic table is unusual, because the second column isn’t really an input, it’s an output. The third column is really the same output, but just the output at a future time.

The D flip flop has two possible values. When D = 0, the flip flop does a reset. A reset means that the output, Q is set to 0. When D = 1, the flip flop does a set, which means the output Q is set to 1.

This is how you can picture the flip flop working. When the clock is not at a positive edge, the flip flop ignores D. However, at the positive edge, it reads in the value, D, and based on D, it updates the value of Q (and of course, Q’).
There is some small amount of delay while it reads in the control input (from D) and the output.

In fact, the “D” in D flip flop stands for “delay”. It basically means that the “D” value is not read immediately, but only at the next positive clock edge.

**T Flip Flop Characteristic Table**

Here’s the characteristic table for a T flip flop.

<table>
<thead>
<tr>
<th>T</th>
<th>Q</th>
<th>Q⁺</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Hold</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Hold</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Toggle</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Toggle</td>
</tr>
</tbody>
</table>

*Fig. 4.11 Truth Table of T Flip flop*

The T flip flop characteristic table has 3 columns. The first column is the value of T, a control input. The second column is the current state, that is, the current value being output by Q. The third column is the next state, that is, the value of Q at the next positive edge. It’s labelled with Q and the superscript, + (the plus sign).

The T flip flop has two possible values. When \( T = 0 \), the flip flop does a hold. A hold means that the output, Q is kept the same as it was before the clock edge. When \( T = 1 \), the flip flop does a toggle, which means the output Q is negated after the clock edge, compared to the value before the clock edge.

Thus, in a T flip flop, you can either maintain the current state’s value for another cycle, or you can toggle the value (negate it) at the next clock edge.

### 4.9 Applications of Flip Flops

In electronics, a **flip-flop** or **latch** is a circuit that has two stable states and can be used to store state information. The circuit can be made to change state by signals applied to one or more control inputs and will have one or two outputs. It is the basic storage element in sequential logic. Flip-flops and latches are a fundamental building block of digital electronics systems used in computers, communications, and many other types of systems.

Flip-flops and latches are used as data storage elements. Such data storage
can be used for storage of state, and such a circuit is described as sequential logic. When used in a finite-state machine, the output and next state depend not only on its current input, but also on its current state (and hence, previous inputs). It can also be used for counting of pulses, and for synchronizing variably-timed input signals to some reference timing signal.

### 4.10 Need for a Register and Types of Registers

Shift register consists of an arrangement of flip-flops and are important in applications involving the storage and transfer data in a digital system, it is a type of sequential logic circuit, mainly for storage of digital data. They are a group of flip-flops connected in a chain so that the output from one flip-flop becomes the input of the next flip-flop. Most of the registers possess no characteristic internal sequence of states. All the flip-flops are driven by a common clock, and all are set or reset simultaneously. Logic circuit shift register

- Since shift register is a device that store data, flip-flop that is suitable to be used is D flip-flop.
- D flip-flop store one (0 or 1) depending on the input data given.
- Since one flip-flop can only store one bit data, to store more than one bit need also more than one flip-flop.

#### Types of Registers

- 4 types of shift register :-
  - Serial in/serial out (SISO)
  - Serial In/Parallel out (SIPO)
  - Parallel in/Serial Out (PISO)
  - Parallel In/Parallel Out (PIPO)

### 4.11 4 Bit Shift Left and Shift Right Register - Working by using 7475

These latches are ideally suited for use as temporary storage for binary information between processing units and inputs/ output or indicator units. Information present at a data (d) input is transferred to the Q output when the enable (C) is high and the Q output will follow the data input as long as the enable remains high. When the enables goes low, the information (that was present at the data input at the time the transition occured) is retained at the Q output until the enable is permitted to go high.
The 75 and LS 75 features complementary Q and Q outputs from a 4 bit latch, and are available in various 16 pin packages. For higher component density applications, the 77 and LS 77 4-bit latches are available in 14 pin flat packages.

These circuits are completely compatible with all popular TTL families. All inputs are diode-clamped to minimize transmission-line effects and simplify system design. Series 54 and 54 LS devices are characterized for operation over the full military temperature range of -55 °C to 125°C, series 74, 74 LS devices are characterized for operation from 0°C to 70°C.

### 4.12 Working of Parallel - In Parallel - Out shift register

The purpose of the parallel-in/parallel-out shift register is to take in parallel data, shift it, then output it as shown below. A universal shift register is a do-everything device in addition to the parallel-in/parallel-out function.

Above we apply four bit of data to a parallel-in/parallel-out shift register at $D_A, D_B, D_C, D_D$. The mode control, which may be multiple inputs, controls parallel loading vs shifting. The mode control may also control the direction of shifting in...
some real devices. The data will be shifted one bit position for each clock pulse. The shifted data is available at the outputs \( Q_A, Q_B, Q_C, Q_D \). The “data in” and “data out” are provided for cascading of multiple stages. Though, above, we can only cascade data for right shifting. We could accommodate cascading of left-shift data by adding a pair of left pointing signals, “data in” and “data out”, above.

Above we apply four bit of data to a parallel-in/parallel-out shift register at \( D_A, D_B, D_C, D_D \). The mode control, which may be multiple inputs, controls parallel loading vs shifting. The mode control may also control the direction of shifting in some real devices. The data will be shifted one bit position for each clock pulse. The shifted data is available at the outputs \( Q_A, Q_B, Q_C, Q_D \). The “data in” and “data out” are provided for cascading of multiple stages. Though, above, we can only cascade data for right shifting. We could accommodate cascading of left-shift data by adding a pair of left pointing signals, “data in” and “data out”, above.

The internal details of a right shifting parallel-in/parallel-out shift register are shown below. The tri-state buffers are not strictly necessary to the parallel-in/parallel-out shift register.

---

### 4.13 Working of Universal Shift Registers - 74194

One of the most common uses of a shift register is to convert between serial and parallel interfaces. This is useful as many circuits work on groups of bits in parallel, but serial interfaces are simpler to construct. Shift registers can be used as simple delay circuits. Several bidirectional shift registers could also be connected in parallel for a hardware implementation of a stack.

SIPO registers are commonly attached to the output of microprocessors when more output pins are required than are available. This allows several binary devices to be controlled using only two or three pins - the devices in question are attached to the parallel outputs of the shift register, then the desired state of all those devices can be sent out of the microprocessor using a single serial connection. Similarly, PISO configurations are commonly used to add more binary inputs to a microprocessor than are available - each binary input (i.e. a switch or button, or more complicated circuitry designed to output high when active) is attached to a parallel input if the shift register, then the data is sent back via serial to the microprocessor using several fewer lines than originally required.

Shift registers can be used also as pulse extenders. Compared to monostable multivibrators, the timing has no dependency on component values, however it requires external clock and the timing accuracy is limited by a
granularity of this clock. Example: Ronja Twister, where five 74164 shift registers create the core of the timing logic this way (schematic).

In early computers, shift registers were used to handle data processing: two numbers to be added were stored in two shift registers and clocked out into an arithmetic and logic unit (ALU) with the result being fed back to the input of one of the shift registers (the accumulator) which was one bit longer since binary addition can only result in an answer that is the same size or one bit longer.

Many computer languages include instructions to ‘shift right’ and ‘shift left’ the data in a register, effectively dividing by two or multiplying by two for each place shifted.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MR</td>
<td>asynchronous master reset (active low)</td>
</tr>
<tr>
<td>2</td>
<td>DS_R</td>
<td>serial data input (shift right)</td>
</tr>
<tr>
<td>3</td>
<td>D_0</td>
<td>parallel data input</td>
</tr>
<tr>
<td>4</td>
<td>D_1</td>
<td>parallel data input</td>
</tr>
<tr>
<td>5</td>
<td>D_2</td>
<td>parallel data input</td>
</tr>
<tr>
<td>6</td>
<td>D_3</td>
<td>parallel data input</td>
</tr>
<tr>
<td>7</td>
<td>D_SL</td>
<td>serial data input (shift left)</td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
<td>ground</td>
</tr>
<tr>
<td>9</td>
<td>S_0</td>
<td>mode control input</td>
</tr>
<tr>
<td>10</td>
<td>S_1</td>
<td>mode control input</td>
</tr>
<tr>
<td>11</td>
<td>CP</td>
<td>clock input (low-to-high, edge-triggered)</td>
</tr>
<tr>
<td>12</td>
<td>Q_3</td>
<td>parallel output</td>
</tr>
<tr>
<td>13</td>
<td>Q_2</td>
<td>parallel output</td>
</tr>
<tr>
<td>14</td>
<td>Q_1</td>
<td>parallel output</td>
</tr>
<tr>
<td>15</td>
<td>Q_0</td>
<td>parallel output</td>
</tr>
<tr>
<td>16</td>
<td>Vcc</td>
<td>supply voltage</td>
</tr>
</tbody>
</table>

Fig. 4.14 Datasheet of IC 74194
Very large serial-in serial-out shift registers (thousands of bits in size) were used in a similar manner to the earlier delay line memory in some devices built in the early 1970s. Such memories were sometimes called circulating memory. For example, the DataPoint 3300 terminal stored its display of 25 rows of 72 columns of upper-case characters using fifty-four 200-bit shift registers, arranged in six tracks of nine packs each, providing storage for 1800 six-bit characters. The shift register design meant that scrolling the terminal display could be accomplished by simply pausing the display output to skip one line of characters.

**Fig. 4.15** IC 74194

### 4.14 Working of Asynchronous Decade Counter

This type of asynchronous counter counts upwards on each leading edge of the input clock signal starting from “0000” until it reaches an output “1010”
(decimal 10). Both outputs QB and QD are now equal to logic “1” and the output from the NAND gate changes state from logic “1” to a logic “0” level and whose output is also connected to the CLEAR (CLR) inputs of all the J-K Flip-flops. This causes all of the Q outputs to be reset back to binary “0000” on the count of 10. Once QB and QD are both equal to logic “0” the output of the NAND gate returns back to a logic level “1” and the counter restarts again from “0000”. We now have a decade or **Modulo-10** counter.

<table>
<thead>
<tr>
<th>Clock Count</th>
<th>QD</th>
<th>QC</th>
<th>QB</th>
<th>QA</th>
<th>Decimal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Counter Resets its output back to zero</td>
</tr>
</tbody>
</table>

**Fig. 4.17 Truth table of Asynchronous decade counter**

### 4.15 Distinguish between Synchronize and Asynchronous Counters

**Comparisions**

**Synchronous Sequential Circuit**: Output changes at discrete interval of time. It is a circuit based on an equal state time or a state time defined by external means such as clock. Examples of synchronous sequential circuit are Flip Flops, Synchronous Counter.

**Asynchronous Sequential Circuit**: Output can be changed at any instant of time by changing the input. It is a circuit whose state time depends solely
upon the internal logic circuit delays. Example of asynchronous sequential circuit is Asynchronous Counter.

### 4.16 Applications of Counters

Counters are one of the many applications of sequential logic that has a widespread use from simple digital alarm clocks to computer memory pointers. A counter is a collection of flip flops, each representing a digit in a binary number representation (which means each bit, depending on position, means a different number).

One of easier ways to build a circuit is to make a flip flop that controls the activation or switching of the second, and so on. This type of counter is called a ripple counter, since the switching signal propagates from one flip flop to the next as in a wave.

### 4.17 List the IC numbers of Flip flops, registers and counters

- 74580: octal transceiver/latch with inverting three-state outputs
- 74589: 8-bit shift register with input latch, three-state outputs
- 74590: 8-bit binary counter with output registers and three-state outputs
- 74592: 8-bit binary counter with input registers
- 74593: 8-bit binary counter with input registers and three-state outputs
- 74594: serial-in shift register with output registers
- 74595: serial-in shift register with output latches
- 74596: serial-in shift register with output registers and open collector outputs
- 74597: serial-out shift register with input latches
- 74598: shift register with input latches
- 74600: dynamic memory refresh controller, transparent and burst modes, for 4K or 16K drams
- 74601: dynamic memory refresh controller, transparent and burst modes, for 64K drams
- 74602: dynamic memory refresh controller, cycle steal and burst modes, for 4K or 16K drams
- 74603: dynamic memory refresh controller, cycle steal and burst modes, for 64K drams
- 74604: octal 2-input multiplexer with latch, high-speed, with three-state outputs
- 74605: latch, high-speed, with open collector outputs
74606 octal 2-input multiplexer with latch, glitch-free, with three-state outputs
74607 octal 2-input multiplexer with latch, glitch-free, with open collector outputs
74608 memory cycle controller
74610 memory mapper, latched, three-state outputs
74611 memory mapper, latched, open collector outputs
74612 memory mapper, three-state outputs
74613 memory mapper, open collector outputs
74620 octal bus transceiver, inverting, three-state outputs
74621 octal bus transceiver, noninverting, open collector outputs
74622 octal bus transceiver, inverting, open collector outputs
74623 octal bus transceiver, noninverting, three-state outputs
74624 voltage-controlled oscillator with enable control, range control, two-phase outputs
74625 dual voltage-controlled oscillator with two-phase outputs
74626 dual voltage-controlled oscillator with enable control, two-phase outputs
74627 dual voltage-controlled oscillator
74628 voltage-controlled oscillator with enable control, range control, external temperature compensation, and two-phase outputs
74629 dual voltage-controlled oscillator with enable control, range control
74630 16-bit error detection and correction (EDAC) with three-state outputs
74631 16-bit error detection and correction with open collector outputs
74632 32-bit error detection and correction
74638 octal bus transceiver with inverting three-state outputs
74639 octal bus transceiver with noninverting three-state outputs
74640 octal bus transceiver with inverting three-state outputs
74641 octal bus transceiver with noninverting open collector outputs
74642 octal bus transceiver with inverting open collector outputs
74643 octal bus transceiver with mix of inverting and noninverting three-state outputs
74644 octal bus transceiver with mix of inverting and noninverting open collector outputs
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>74645</td>
<td>octal bus transceiver</td>
</tr>
<tr>
<td>74646</td>
<td>octal bus transceiver/latch/multiplexer with noninverting three-state outputs</td>
</tr>
<tr>
<td>74647</td>
<td>octal bus transceiver/latch/multiplexer with noninverting open collector outputs</td>
</tr>
<tr>
<td>74648</td>
<td>octal bus transceiver/latch/multiplexer with inverting three-state outputs</td>
</tr>
<tr>
<td>74649</td>
<td>octal bus transceiver/latch/multiplexer with inverting open collector outputs</td>
</tr>
<tr>
<td>74651</td>
<td>octal bus transceiver/register with inverting three-state outputs</td>
</tr>
<tr>
<td>74652</td>
<td>octal bus transceiver/register with noninverting three-state outputs</td>
</tr>
<tr>
<td>74653</td>
<td>octal bus transceiver/register with inverting three-state and open collector outputs</td>
</tr>
<tr>
<td>74654</td>
<td>octal bus transceiver/register with noninverting three-state and open collector outputs</td>
</tr>
<tr>
<td>74658</td>
<td>octal bus transceiver with Parity, inverting</td>
</tr>
<tr>
<td>74659</td>
<td>octal bus transceiver with Parity, noninverting</td>
</tr>
<tr>
<td>74664</td>
<td>octal bus transceiver with Parity, inverting</td>
</tr>
<tr>
<td>74665</td>
<td>octal bus transceiver with Parity, noninverting</td>
</tr>
<tr>
<td>74668</td>
<td>synchronous 4-bit decade Up/down counter</td>
</tr>
<tr>
<td>74669</td>
<td>synchronous 4-bit binary Up/down counter</td>
</tr>
<tr>
<td>74670</td>
<td>4 by 4 register File with three-state outputs</td>
</tr>
<tr>
<td>74671</td>
<td>4-bit bidirectional shift register/latch/multiplexer with three-state outputs</td>
</tr>
<tr>
<td>74672</td>
<td>4-bit bidirectional shift register/latch/multiplexer with three-state outputs</td>
</tr>
<tr>
<td>74673</td>
<td>16-bit serial-in serial-out shift register with output storage registers, three-state outputs</td>
</tr>
<tr>
<td>74674</td>
<td>16-bit parallel-in serial-out shift register with three-state outputs</td>
</tr>
<tr>
<td>74677</td>
<td>16-bit address comparator with enable</td>
</tr>
<tr>
<td>74678</td>
<td>16-bit address comparator with latch</td>
</tr>
<tr>
<td>74679</td>
<td>12-bit address comparator with latch</td>
</tr>
<tr>
<td>74680</td>
<td>12-bit address comparator with enable</td>
</tr>
<tr>
<td>74681</td>
<td>4-bit parallel binary accumulator</td>
</tr>
<tr>
<td>74682</td>
<td>8-bit magnitude comparator</td>
</tr>
<tr>
<td>74683</td>
<td>8-bit magnitude comparator with open collector outputs</td>
</tr>
<tr>
<td>Part Numbers</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>74684</td>
<td>8-bit magnitude comparator</td>
</tr>
<tr>
<td>74685</td>
<td>8-bit magnitude comparator with open collector outputs</td>
</tr>
<tr>
<td>74686</td>
<td>8-bit magnitude comparator with enable</td>
</tr>
<tr>
<td>74687</td>
<td>8-bit magnitude comparator with enable</td>
</tr>
<tr>
<td>74688</td>
<td>8-bit equality comparator</td>
</tr>
<tr>
<td>74689</td>
<td>8-bit magnitude comparator with open collector outputs</td>
</tr>
<tr>
<td>74690</td>
<td>Three state outputs</td>
</tr>
<tr>
<td>74691</td>
<td>4-bit binary counter/latch/multiplexer with asynchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74692</td>
<td>4-bit decimal counter/latch/multiplexer with synchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74693</td>
<td>4-bit binary counter/latch/multiplexer with synchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74694</td>
<td>4-bit decimal counter/latch/multiplexer with synchronous and asynchronous resets, three-state outputs</td>
</tr>
<tr>
<td>74695</td>
<td>4-bit binary counter/latch/multiplexer with synchronous and asynchronous resets, three-state outputs</td>
</tr>
<tr>
<td>74696</td>
<td>4-bit decimal counter/register/multiplexer with asynchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74697</td>
<td>4-bit binary counter/register/multiplexer with asynchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74698</td>
<td>4-bit decimal counter/register/multiplexer with synchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74699</td>
<td>4-bit binary counter/register/multiplexer with synchronous reset, three-state outputs</td>
</tr>
<tr>
<td>74716</td>
<td>Programmable decade counter</td>
</tr>
<tr>
<td>74718</td>
<td>Programmable binary counter</td>
</tr>
<tr>
<td>74724</td>
<td>Voltage controlled multivibrator</td>
</tr>
<tr>
<td>74740</td>
<td>Octal buffer/Line driver, inverting, three-state outputs</td>
</tr>
<tr>
<td>74741</td>
<td>Octal buffer/Line driver, noninverting, three-state outputs, mixed enable polarity</td>
</tr>
<tr>
<td>74744</td>
<td>Octal buffer/Line driver, noninverting, three-state outputs</td>
</tr>
<tr>
<td>74748</td>
<td>8 to 3-line priority encoder</td>
</tr>
<tr>
<td>74779</td>
<td>8-bit bidirectional binary counter (3-state)</td>
</tr>
<tr>
<td>74783</td>
<td>Synchronous address multiplexer</td>
</tr>
<tr>
<td>74790</td>
<td>Error detection and correction (EDAC)</td>
</tr>
<tr>
<td>74794</td>
<td>8-bit register with readback</td>
</tr>
<tr>
<td>74795</td>
<td>Octal buffer with three-state outputs</td>
</tr>
<tr>
<td>74796</td>
<td>Octal buffer with three-state outputs</td>
</tr>
<tr>
<td>74797</td>
<td>Octal buffer with three-state outputs</td>
</tr>
</tbody>
</table>
74798 octal buffer with three-state outputs
74804 hex 2-input NAND drivers
74805 hex 2-input NOR drivers
74808 hex 2-input AND drivers
74832 hex 2-input OR drivers
74848 8 to 3-line priority encoder with three-state outputs
74873 octal transparent latch
74874 octal d-type flip-flop
74876 octal d-type flip-flop with inverting outputs
74878 dual 4-bit d-type flip-flop with synchronous clear, noninverting three-state outputs
74879 dual 4-bit d-type flip-flop with synchronous clear, inverting three-state outputs
74880 octal transparent latch with inverting outputs
74881 arithmetic logic unit
74882 32-bit lookahead carry generator
74888 8-bit slice processor
74901 hex inverting TTL buffer
74902 hex non-inverting TTL buffer
74903 hex inverting CMOS buffer
74904 hex non-inverting CMOS buffer
74905 12-Bit successive approximation register
74906 hex open drain n-channel buffers
74907 hex open drain p-channel buffers
74908 dual CMOS 30V relay driver
74909 quad voltage comparator
74910 256x1 CMOS static RAM
74911 4-digit expandable display controller
74912 6-digit BCD display controller and driver
74914 hex schmitt trigger with extended input voltage
74915 seven segment to BCD decoder
74917 6-digit Hex display controller and driver
74918 dual CMOS 30V relay driver
74920 256x4 CMOS static RAM
74921 256x4 CMOS static RAM
74922 16-key encoder
74923 20-key encoder
74925 4-digit counter/display driver
74926 4-digit counter/display driver
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>74927</td>
<td>4-digit counter/display driver</td>
</tr>
<tr>
<td>74928</td>
<td>4-digit counter/display driver</td>
</tr>
<tr>
<td>74929</td>
<td>1024x1 CMOS static RAM</td>
</tr>
<tr>
<td>74930</td>
<td>1024x1 CMOS static RAM</td>
</tr>
<tr>
<td>74932</td>
<td>phase comparator</td>
</tr>
<tr>
<td>74933</td>
<td>address bus comparator</td>
</tr>
<tr>
<td>74934</td>
<td>=ADC0829 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74935</td>
<td>3.5-digit digital voltmeter (DVM) support chip for multiplexed 7-segment displays</td>
</tr>
<tr>
<td>74936</td>
<td>3.75-digit digital voltmeter (DVM) support chip for multiplexed 7-segment displays</td>
</tr>
<tr>
<td>74937</td>
<td>=ADC3511 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74938</td>
<td>=ADC3711 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74941</td>
<td>octal bus/line drivers/line receivers</td>
</tr>
<tr>
<td>74945</td>
<td>4 digit up/down counter with decoder and driver</td>
</tr>
<tr>
<td>74947</td>
<td>4 digit up/down counter with decoder and driver</td>
</tr>
<tr>
<td>74948</td>
<td>=ADC0816 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74949</td>
<td>=ADC0808 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>74949</td>
<td>=ADC0808 ADC, see corresponding NSC datasheet</td>
</tr>
<tr>
<td>741005</td>
<td>hex inverting buffer with open-collector output</td>
</tr>
<tr>
<td>741035</td>
<td>hex noninverting buffers with open-collector outputs</td>
</tr>
<tr>
<td>742960</td>
<td>error detection and correction (EDAC)</td>
</tr>
<tr>
<td>742961</td>
<td>edac bus buffer, inverting</td>
</tr>
<tr>
<td>742962</td>
<td>edac bus buffer, noninverting</td>
</tr>
<tr>
<td>742968</td>
<td>dynamic memory controller</td>
</tr>
<tr>
<td>742969</td>
<td>memory timing controller for use with EDAC</td>
</tr>
<tr>
<td>742970</td>
<td>memory timing controller for use without EDAC</td>
</tr>
<tr>
<td>741G3208</td>
<td>single 3 input OR-AND Gate;</td>
</tr>
<tr>
<td>744002</td>
<td>dual 4-input NOR gate</td>
</tr>
<tr>
<td>744015</td>
<td>dual 4-bit shift registers</td>
</tr>
<tr>
<td>744017</td>
<td>5-stage ÷10 Johnson counter</td>
</tr>
<tr>
<td>744020</td>
<td>14-stage binary counter</td>
</tr>
<tr>
<td>744024</td>
<td>7 stage ripple carry binary counter</td>
</tr>
<tr>
<td>744028</td>
<td>BCD to decimal decoder</td>
</tr>
<tr>
<td>744040</td>
<td>12-stage binary ripple counter</td>
</tr>
<tr>
<td>744046</td>
<td>phase-locked loop and voltage-controlled oscillator</td>
</tr>
<tr>
<td>744049</td>
<td>hex inverting buffer</td>
</tr>
<tr>
<td>744050</td>
<td>hex buffer/ converter (non-inverting)</td>
</tr>
<tr>
<td>744051</td>
<td>high-speed CMOS 8-channel analog multiplexer/</td>
</tr>
</tbody>
</table>
Short Answer Type Questions

1. What is sequential logic circuit.
2. Draw the NAND gate.
3. Draw the NOR gate.
4. Draw the clock pulse.
5. What is level triggering?
6. What is edge triggering?
7. Draw the clocked SR flip flop.
8. What preset and clear inputs?
10. Draw the D flip flop.
11. Draw the T flip flop.
12. Write applications of flip flop.
13. What are the types of registers?
14. Mention the number of pins in IC 7475.
15. What are the applications of universal shift register?
16. What are the applications of asynchronous counters.
17. What are the applications of synchronous counters?
18. What are the applications of counters?

**Long Answer Type Questions**

1. Draw and explain working of NAND gate with truth table.
2. Draw and explain working of NOR gate with truth table.
3. Explain working of level/edge triggering.
4. Draw and explain working of SR flip flop using NAND gate.
5. Explain need of clear and preset inputs.
7. Explain working of D,T flip flops with truth table.
8. Explain the 4 bit shift left, shift right register with neat diagram.
10. Explain working of universal shift register.
11. Draw and explain working of asynchronous decade counters.
12. Draw and explain working of synchronous.

**On Job Training / Practical Questions**

1. Study the NAND, NOR latches with truth tables.
2. Study the level/edge triggering with clock pulse.
3. Study the SR flip flop using NAND gates.
4. Study the D,T flip flops with truth table.
5. Study the shift register and types.
6. Study the universal shift register.
7. Study the asynchronous counters.
8. Study the IC’s used in flip flops, registers, counters.
Learning Objectives

After studying this unit, the student will be able to

• To study the need of A/D and D/A converters.

• To define resolution, accuracy, monotonicity and setting time of D/A converters.

• To study the working of D/A conversion using binary weighted resistors.

• To study the working of D/A conversion using R-2R ladder network.

• To study the working of A/D conversion using counter method.

• To study the A/D conversion using successive approximation method.

5.0 Introduction

Digital systems are widely used in many applications such as communication, control systems, computers, instrumentations, etc. In many of these applications, the signals are available only in analog form, but not in digital form. Hence by using digital hardware, these analog signals have to be converted into digital form. The process of converting analog signals into digital form is known as analog to digital conversion (A/D). The system which is used for this process is known as ‘A/D converter (ADC). The process of converting digital signal into analog signal...
Electronics Engineering Technician

voltage or current. This process is known as ‘Digital to analog conversion’ (D/A) and the system used for this purpose is known as digital to analog D/A converter (DAC).

5.1 Need for A/D and D/A Conversion

In many of the applications digital systems must be interfaced with analog equipment. In such cases, the digital system is needed for A/D and D/A converters; for the communications or the data transfer with analog equipment.

1) In the fig. 2.1 shows a typical situation in which the digital system has analog inputs and outputs.

The input section is a variable voltage source, giving a continuous voltage ranging from 0 to 10 V. Then the A/D converter translates the analog input digital data. After the computer of processing the data in the digital system, the system gives the digital 1 data output. This output will be translated into an analog voltage by the D/A converter. This output analog voltage is indicated by the voltmeter.

Fig. 5.1 Digital System with A/D and D/A converters

Thus the above situation necessitate A/D and D/A converters along with the digital system.

2) Similarly, a digital communication system is used to transmit to signals, which are in the form of electric signals. This necessitate or requires an D/A converter at the transmitting and a A/D converter at the receiving end.

Methods of A/D and D/A Conversions:

a) Digital to Analog (D/A) Conversion:

The process of converting a digital signal into analog is known as ‘Digital-to-analog’ (D/A) conversion. There are two types of D/A converters 1) Weighted-Register D/A converter 2) R-2 R circuit i.e., D/A converter.

Basic Principle of D/A Conversion

The basic principle of operation of D/A conversion is shown in the block diagram of a D/A converter in the below figure.

Counter type A/D Converter

The black diagram of a counter type 4 bit A/D converter is shown in the fig below. It employs a voltage comparator, an and gate a BCD counter and a D/A converter.
5.2 Terms Explanation

5.2.1 Voltage Resolution

Voltage resolution = \((\text{voltage span}) / 2^N\)

Conversely, the minimum required number of bits can be determined by solving the above equation for \(N\) as shown belw. Any fractional part of \(N\) is rounded up to the next higher integer.

\[ N = \log_2 \left( \frac{\text{voltage span}}{\text{voltage resolution}} \right) \]

As an example, an 8-bit A/D whose voltage span is 10 volts has a voltage resolution of 39 milli volts. If the voltage resolution of 1 mV is needed for a 6 volt converter then the required number of bits is 13.

An example, the maximum output voltage of an 8-bit unipolar D/A whose voltage span is 10 volts is 0 +0 -0.039 = 9.961 volts. If the converter were bipolar then the maximum output voltage would be -5 +10 - 0.039 = 4.961 volts.

5.2.2 Accuracy

Accuracy refers to the accuracy of the voltage span. Typical values are in the 0.5% range.

5.2.3 Monotonicity

Monotonicity refers to whether the output conversion magnitude always increases for an increase in the input magnitude. Most converters are guaranteed to be monotonic over some specified operating conditions. If you operate them outside that range then the output may fail a monotonicity test at some levels. This could be serious in servo systems as monotonic failure is a phase inversion and a limit cycle oscillation becomes possible.

Linearity

Linearity refers to the maximum deviation of the output from a theoretically perfect straight line. Inexpensive converters may have linearity errors of several LSBs. Premium converters are often linear to less than one LSB.

Settling Time

The **settling time** of an amplifier or other output device is the time elapsed from the application of an ideal instantaneous step input to the time at which the amplifier output has entered and remained within a specified error band, usually symmetrical about the final value. Settling time includes a very brief propagation
delay, plus the time required for the output to slew to the vicinity of the final value, recover from the overload condition associated with slew, and finally settle to within the specified error.

5.3 D/A Conversion by using Binary Weighted Resistors

A D/A converter using binary-weighted resistors is shown in the figure below. In the circuit, the op-amp is connected in the inverting mode. The op-amp can also be connected in the non-inverting mode. The circuit diagram represents a 4-digit converter. Thus, the number of binary inputs is four.

![Fig. 5.1 Binary weighted resistors D/A conversion](image)

We know that, a 4-bit converter will have \(2^4 = 16\) combinations of output. Thus, a corresponding 16 outputs of analog will also be present for the binary inputs.

Four switches from b0 to b3 are available to simulate the binary inputs: in practice, a 4-bit binary counter such as a 7493 can also be used.

**Working**

The circuit is basically working as a current to voltage converter.

- b0 is closed

It will be connected directly to the +5V.

Thus, voltage across R = 5V

Current through R = \(5V/10\text{ kohm} = 0.5\text{ mA}\)

Current through feedback resistor, \(R_f = 0.5\text{ mA}\) (Since, Input bias current, \(I_B\) is negligible)
Thus, output voltage = -(1kohm)*(0.5mA) = -0.5V

· b1 is closed, b0 is open

R/2 will be connected to the positive supply of the +5V.

Current through R will become twice the value of current (1mA) to flow through Rf.

Thus, output voltage also doubles.

· b0 and b1 are closed

Current through Rf = 1.5mA

Output voltage = -(1kohm)*(1.5mA) = -1.5V

Thus, according to the position (ON/OFF) of the switches (b0-b3), the corresponding “binary-weighted” currents will be obtained in the input resistor. The current through Rf will be the sum of these currents. This overall current is then converted to its proportional output voltage. Naturally, the output will be maximum if the switches (b0-b3) are closed

\[ V_0 = -R_f \times \left[ \frac{b_3}{R} \frac{b_2}{R/2} \frac{b_1}{R/4} \frac{b_0}{R/8} \right] \]

– where each of the inputs b3, b2, b1, and b0 may either be HIGH (+5V) or LOW (0V).

The graph with the analog outputs versus possible combinations of inputs is shown below

![Graphical representation of D/A conversion by Binary weighted resistors](image)

Fig. 5.2 Graphical representation of D/A conversion by Binary weighted resistors

The output is a negative going staircase waveform with 15 steps of -0.5V each. In practice, due to the variations in the logic HIGH voltage levels, all the steps will not have the same size. The value of the feedback resistor Rf changes
the size of the steps. Thus, a desired size for a step can be obtained by connecting the appropriate feedback resistor. The only condition to look out for is that the maximum output voltage should not exceed the saturation levels of the op-amp. Metal-film resistors are more preferred for obtaining accurate outputs.

**Disadvantages**

If the number of inputs (>4) or combinations (>16) is more, the binary-weighted resistors may not be readily available. This is why; R and 2R method is more preferred as it requires only two sets of precision resistance values.

### 5.4 R-2R Ladder Network: Working

A D/A converter with R and 2R resistors is shown in the figure below. As in the binary-weighted resistors method, the binary inputs are simulated by the switches (b0-b3), and the output is proportional to the binary inputs. Binary inputs can be either in the HIGH (+5V) or LOW (0V) state. Let b3 be the most significant bit and thus is connected to the +5V and all the other switches are connected to the ground.

![Fig. 5.3 R-2R Ladder Network D/A conversion](image)

\[
R_m = [(2RI_{2R} + R) II_{2R}] + R = 2R = 20\text{kOhms.}
\]

The resultant circuit is shown below.

In the figure shown above, the negative input is at virtual ground, therefore the current through \( R_m \) = 0.

Current through 2R connected to +5V = 5V/20kohm = 0.25 mA

The current will be the same as that in Rf.

\[
V_o = -(20\text{kohm})*(0.25\text{mA}) = -5V
\]
Output voltage equation is given below.

\[ V_0 = -Rf \left( b_3/2R + b_2/4R + b_1/8R + b_0/16R \right) \]

### 5.5 A/D Conversion by Counter Method Working

**A/D, D/A converters**

Block diagram of a counter type 4-bit aid converter

![Block diagram of A/D conversion by counter method](image)

**Fig. 5.5 A/D Conversion Counter Method**
Operation

Let the counter begin RESET and the output of the D/A converter is zero. Apply the analog voltage at the input. If the analog input voltages at A is greater than the voltage at the input B of the comparator, then the output of the comparator switches to a high state and enables the AND gate. The clock pulse is allowed to increase or advance the count of the BCD counter, through its binary states. The counter continues to advance from one binary state to the next producing successively higher steps and the count will be displayed at the digital or binary outputs as D D D D. The count on the counter increases until the feedback voltage from D/A converter increases becomes greater than the analog input voltage. Whenever the voltage at B is greater than at A, the comparator output will go low and disable the AND gate. It results the cutting off the clock pulses to stop the counter. The state of the counter at this point equals the number of steps in the reference voltage or feed back voltage at which the comparison occurs voltage. Note that for each sample of analog voltage, the counter must count from zero up to point at which the feedback voltage reaches the analog input voltage.

Thus A/D converter produces the digital output of an analog input voltage.

Disadvantages

1. Slow speed of operation is the main drawback of this method. In the worst case of maximum input, the counter must sequence through its maximum number of states before the conversion occurs. For a 4-bit, 8-bit and 16-bit conversion, this means a maximum of 16, 256 and 4096 counter states are required respectively.

2. For each sample of analog voltage, the counter must count from zero up to the point at which the feedback voltage reached the analog input voltage.

3. The conversion time varies, depending upon the analog voltage.

5.6 A/D Conversion by Successive Approximation Method

This is the most common form of conversion and uses a D/A converter and an algorithm to converge to the digital representation of the analog input voltage. The algorithm begins with all bits to the D/A converter set to 0 except the most significant bit. If the analog input voltage is higher than the D/A output voltage then the D/A bit is latched to a ‘1’.

Then the next least bit of the D/A converter is set to ‘1’ and the resulting output voltage is compared to the input voltage. If the input voltage is higher
then this D/A bit is also latched to ‘1’. This algorithm continues all the way to the least significant bit. It is important for the input voltage to remain stable during the conversion process. Thus, a analog input. Figure 2 shows the basic structure

As an example of operation the process of digitizing a 6.9 will be illustrated using a unipolar 8 - bit converter with a maximum voltage of 10.24 (this value is chosen so as to make simple fractional voltages in the example). The conversion process begins with a START signal. The TRAK-HOLD switch then opens so that the signal input to the comparator is constant during the conversion process. The state machine then sets the digital signal to the D/A to binary 10000000 for an output of 5.12 volts. The comparator output is ‘1’ since 6.9 is greater than 5.12. So, the most significant bit is latched. Next, the state machine sets the digital signal to the D/A converter to binary 11000000 for an output of 7.68 volts. The comparator output is ‘0’ since 6.9 is less than 7.68. So, this bit of the D/A number is latched to ‘0’.

- **DAC** = Digital - to - Analog Converter.
- **EOC** = End of Conversion.
- **SAR** = Successive Approximation Register.
- **S/H** = Sample and Hold circuit.
- **V_{in}** = input voltage.
- **V_{ref}** = reference voltage

**Fig. 5.6 A/D Conversion by Successive approximation method**
Short Answer Type Questions

1. Expand A/D, D/A converters.
2. Define resolution.
3. Define accuracy.
4. Define monotonicity.
5. Define setting time.

Long Answer Type Questions

1. Explain the need A/D and D/A converters.
2. Explain the working of D/A conversion by using binary weighted resistors method.
3. Explain the working of D/A conversion R-2R method.
4. Explain the working of A/D conversion using counter method.
5. Explain A/D conversion using successive approximation method.

On Job Training

1. Study the A/D, D/A conversion binary weighed resister method.
2. Study the D/A working of R-2R ladder network.
3. Study the A/D conversion using successive approximation method.
6.0 Introduction to Semiconductor Memories

Memory Terminology and its General Operations

One of the major advantages that digital systems have over analog systems is the ability to easily store large quantities of digital information and data for short or long periods of time. This memory capability is what makes digital
computer so versatile and adaptable to many situations. For example, in a digital computer the internal main memory stores instructions that tell the computer what to do under all the possible circumstances so that the computer will do its job with a minimum amount of human intervention. Magnetic tape and magnetic disk are popular mass storage devices that are much less and expensive in cost per bit than internal memory devices.

### 6.1 Memory Terminology

**Memory Cell:** This is a device or electrical circuit used to store a single bit 0 or 1 is known as memory cell.

**Memory Word:** This is a group of bits (cells) in a memory that represents information or data of some type is known as memory word. The word size in modern computers typically ranges from 4 to 64 bits, depending on the size of the computer.

**Byte:** This is a special term used word length is known as byte. In modern digital computers, the word size is usually a multiple of 8-bits such as 2 bytes or 4 bytes.

**Memory Capacity:** Computer memory is measured in terms of bytes. The larger units are Kilobyte, Megabyte, Gigabyte, Terabyte, Petabyte, Exabyte, Zettabyte and Yottabyte.

- 1000 bytes = 1 Kilo byte
- 1000 Kilobyte = 1 Megabyte
- 1000 Megabyte = 1 Gigabyte
- 1000 Gigabyte = 1 Terabyte
- 1000 Terabyte = 1 Petabyte
- 1000 Petabyte = 1 Exabyte
- 1000 Exabyte = 1 Yottabyte

**Address:** This is a number that identifies the location of word in memory is known as address. Each word stored in a memory device or system has unique address. Addresses are always specified as a binary number, although octal, hexadecimal and decimal numbers are often used for convenience.

**Read Operation:** This is the operation whereby the binary word stored in a specific memory location (address) is sensed and then transferred to another location is known as read operation. For example, if we want to use ‘word 42
of the memory above image for some purpose, we have to perform a read operation on address 100. The read operation is often called a fetch operation, since a word is being fetched from memory. We use both terms interchangeable.

**Write Operation:** This is the operation whereby a new word is placed into a particular memory location is known as write operation. It is also referred to as a store operation. Whenever a new word is written into a memory location, it replaces the word that was previously stored there. The old is lost in the process of writing into this memory location.

**Access Time:** It is the measure of a memory device’s operating speed is known as access time. It is the amount of time required to perform a read operation. More specifically, it is the time between the memory receiving a read command signal and the data becoming available at the memory output.

**Cycle Time:** It is the amount of time required for the memory to perform a read or write operation and then return to its original state ready for the next operation is known as cycle time. Cycle time is normally longer than access time.

**Random Access Memory (RAM):** This refers to memories in which the actual physical location of a memory word has no effect on how long it takes to read from or write into that location is known as Random Access Memory (RAM). In other words, the access time is the same for any address in memory. Most semiconductor memories and magnetic disk memories are random access memories.

**Sequential Access Memory** It is a type of memory in which the access time is not constant but varies depending on the address location is known as sequential access memory. A particular sorted word is found by sequencing through all address locations until the desired address is reached. This produces access time, which are much longer than those of random access memories. Example of sequential access memory devices includes magnetic tapes.

**Read/Write Memory (RWM):** It is any memory that can be read from or written into with equal case is known as read/write memory.

**Read-Only Memory (ROM):** It refers to a broad class of semiconductor memories designed for applications where the ratio of read operation to write operations is very high is known as read only memory. Technically, a ROM can be written into (programmed) only once and this operation normally performed at the factory. Thereafter information can only be read from the memory. Other types of ROM are actually read-mostly memories more complicated than the read operation. Therefore the write operation is not performed very often.
Static Memory Devices: These are those semiconductor memory devices in which the stored data will remain permanently stored as long as power is supplied devices without the need for periodically rewriting the data into memory is known as static memory devices.

Dynamic Memory Devices: These are those semiconductor memory devices in which the stored data will not remain permanently stored, even with power applied, unless the data is periodically rewritten into memory. This operation is called a refresh operation.

General Memory Operations

Although each type of memory is different in its internal operation, there are certain basic operating principles that are the same for all memory systems. Every memory system requires several different types of input and output lines to perform the following functions.

1. Select the address in memory that is being accessed for a read or write operation.
2. Select either a read or write operation to be performed.
3. Supply the input data to be stored in memory during a write operation.
4. Hold the output data coming from memory during read operation.
5. Enable (or disable) the memory so that it will

6.2 Classification of Memories

Memories are two types.

1. Random Access Memory (RAM)
2. Read Only Memory (ROM)

Types of RAM

1. SRAM
2. DRAM
3. VRAM
4. NVRAM
5. MOS RAM
Types of ROM

1. ROM
2. PROM
3. EPROM
4. EEPROM

Internal storage areas in the computer. The term memory identifies data storage that comes in the form of chips, and the word storage is used for memory that exists on tapes or disks.

Moreover, the term memory is usually used as shorthand for physical memory, which refers to the actual chips capable of holding data. Some computers also use virtual memory, which expands physical memory onto a hard disk.

Every computer comes with a certain amount of physical memory, usually referred to as main memory or RAM. You can think of main memory as an array of boxes, each of which can hold a single byte of information. A computer that has 1 megabyte of memory, therefore, can hold about 1 million bytes (or characters) of information.

Memory Manufacturers Corsair, Kingston, Rambus, Crucial, Samsung, PNY, Micron

Memory Terminology

Memory speed is measured in nanoseconds, this is the time to access data that is stored in memory the lower the nanoseconds the faster the memory (2ns-80ns)
Fig. 6.2 Types of Memories

**Memory**

- **ROM** - Read Only Memory
  - PROM - Programmable Read Only Memory
  - EPROM - Erasable Programmable Read Only Memory
  - EEPROM - Electrically Erasable Programmable Read Only Memory

- **RAM** - Random Access Memory
  - SRAM - Static RAM
  - DRAM - Dynamic RAM
  - CACHE - (L1, L2, L3)

**DRAM Packages**

- **SO DIMM** (72, 100, 144, 200, 240-pin)
  - Laptops
  - (SDDRAM or RIMM-SDDRAM)
  - (DDR2-SDDRAM)
  - (DDR3-SDDRAM)

- **Micro DIMM** (172, 21+pin)
  - (FPM or EDO)
    - (Fast Page Mode or EDO)

- **30-pin SIIIMM**
  - (Extended Data Out)

- **72-pin SIIIMM**
  - (SDDRAM)
    - (Synchronous Dynamic RAM)

- **184-pin SIIIMM**
  - (DDR-SDDRAM)
    - (Double Data Rate SDRAM)

- **240-pin SIIIMM**
  - (DDR2-SDDRAM)
    - (Double Data Rate 2 SDRAM)

- **184-pin RIMM**
  - (RDRAM)
    - (Rambus Dynamic RAM)

- **232-pin RIMM**
  - (RDRAM)
    - (Rambus Dynamic RAM)
**Parity** - A simple error checking method used in memory correction is known as parity.

**ECC** - Error Correction Code used in memory correction for new generation computers.

**Memory Banks** - A socket where memory is installed is known as memory banks. Example 4 banks will have 8MB of memory each for a total of 32MB.

**There are several different types of memory:**

**SDRAM**

![Fig. 6.4 SDRAM](image)

**ROM or Read Only Memory**: The computers almost always contain a small amount of read-only memory that holds instructions for starting up the computer. Unlike RAM, ROM cannot be written to. It is non-volatile which means once you turn off the computer the information is still there.

![Fig. 6.5 ROM](image)

**PROM**: It is read as programmable read-only memory. A PROM is a memory chip on which data can be written only once. Once a program has been written onto a PROM, it remains there forever. Unlike RAM, PROM’s retain their contents when the computer is turned off. The difference between a PROM and a ROM (read-only memory) is that a PROM is manufactured as blank.
Electronics Engineering Technician

memory, whereas a ROM is programmed during the manufacturing process. To write data onto a PROM chip, you need a special device called a PROM programmer or PROM burner. The process of programming a PROM is sometimes called burning the PROM.

**EPROM:** It is read as Erasable Programmable Read-Only Memory. This is a special type of PROM that can be erased by exposing it to ultraviolet light. Once it is erased, it can be reprogrammed. An EEPROM is similar to a PROM, but requires only electricity to be erased.

**EEPROM:** A electrically erasable programmable read only memory, Pronounced double-ee-prom or e-e-prom, an EEPROM is a special type of PROM that can be erased by exposing it to an electrical charge. Like other types of PROM, EEPROM retains its contents even when the power is turned off. Also like other types of ROM, EEPROM is not as fast as RAM. EEPROM is similar to flash memory (sometimes called flash EEPROM). The principal difference is that EEPROM requires data to be written or erased one byte at a time whereas flash memory allows data to be written or erased in blocks.

**RAM (Random Access Memory)** is a temporary (Volatile) storage area utilized by the CPU. Before a program can be ran the program is loaded into the memory which allows the CPU direct access to the program.

**SRAM** Short for static random access memory, and pronounced ess-ram. SRAM is a type of memory that is faster and more reliable than the more common DRAM (dynamic RAM). The term static is derived from the fact that it doesn’t need to be refreshed like dynamic RAM.

SRAM is often used only as a **memory cache** usually found in the CPU (L1, L2 and L3 Cache)

**DRAM** stands for dynamic random access memory, a type of memory used in most personal computers.

**Types of DRAM Packages and DRAM Memory**

![SO-DIMM](image-url)
SO-DIMM Short for Small Outline DIMM, a small version of a DIMM used commonly in notebook computers. 72 supports 32bit and 144 and 200 SO-DIMM pins support a full 64-bit transfer.

![Image of SO-DIMM](image1)

**Fig. 6.7 (144, 172) Micro-DIMM**

Micro-DIMM Short for Micro Dual Inline Memory Module, a competing memory used on laptops, mostly supports 144 and 172 pins.

SIMM Acronym for single in-line memory module, a small circuit board that can hold a group of memory chips. Typically, SIMM’s holds up 8 (on Macintoshes) or 9 (on PCs) RAM chips. On PCs, the ninth chip is often used for parity error checking. Unlike memory chips, SIMM’s is measured in bytes rather than bits. SIMM’s is easier to install than individual memory chips. A SIMM is either 30 or 72 pins.

![Image of SIMM](image2)

**Fig. 6.8 30 pin SIMM (Usually FPM or EDO RAM)**

FPM RAM Short for Fast Page Mode RAM, a type of Dynamic RAM (DRAM) that allows faster access to data in the same row or page. Page-mode memory works by eliminating the need for a row address if data is located in the row previously accessed. It is sometimes called page mode memory.

![Image of FPM RAM](image3)

**Fig. 6.9 72 pin SIMM (EDO RAM)**
**EDO DRAM** Short for Extended Data Output Dynamic Random Access Memory, a type of DRAM that is faster than conventional DRAM. Unlike conventional DRAM which can only access one block of data at a time, EDO RAM can start fetching the next block of memory at the same time that it sends the previous block to the CPU.

**DIMM** Short for dual in-line memory module, a small circuit board that holds memory chips. A single in-line memory module (SIMM) has a 32-bit path to the memory chips whereas a DIMM has 64-bit path. Because the Pentium processor requires a 64-bit path to memory, you need to install SIMM’s two at a time. With DIMM’s, you can install memory one DIMM at a time. A DIMM contains 168 pins.

![168 pin DIMM (SDRAM)](image)

**SDRAM** Short for Synchronous DRAM, a new type of DRAM that can run at much higher clock speeds than conventional memory. SDRAM actually synchronizes itself with the CPU’s bus and is capable of running at 133 MHz, about three times faster than conventional FPM RAM, and about twice as fast EDO DRAM. SDRAM is replacing EDO DRAM in many newer computers.

SDRAM delivers data in high speed burst.

![184 pin DIMM (DDR-SDRAM)](image)

**DDR SDRAM** Short for Double Data Rate-Synchronous DRAM, a type of SDRAM that supports data transfers on both edges of each clock cycle, effectively doubling the memory chip’s data throughput. DDR-SDRAM is also called SDRAM II.
Fig. 6.12 240 DIMM (DDR2-SDRAM)

**DDR2-SDRAM** Short for Double Data Rate Synchronous DRAM 2 is a type of DDR that supports higher’s speeds than it’s predecessor DDR SDRAM.

Fig. 6.13 240 DIMM (DDR3-SDRAM)

---

### 6.3 Differentiate between ROM and RAM

RAM is short for Random Access Memory, and comes from hardware components wired into or attached to the motherboard, the main circuit board of your computer. RAM is used to run certain basic programs and functions that your computer needs to operate correctly, and functions only while the computer is receiving power. Programs you’re using are written in RAM temporarily while the computer is processing them. Think of RAM as a playing field, a large open area where your programs function. Each program takes up a certain amount of space; the field can accommodate one or several different programs at one time, but its capacity is limited. When you shut down a program, it disappears from RAM and (ideally) the space it occupied can be reused. Sometimes some operating systems, including Microsoft Windows, won’t relinquish the RAM space even when a program is closed. However, because stuff in the RAM is retained only while the computer is powered up, turning it off will always clear the RAM. ROM is an acronym for Read Only Memory, a type of unchangeable memory residing in chips on your motherboard. ROM contains the bare minimum of instructions needed to start your computer. Because it’s used for critical functions, it can’t be removed short of ripping it out of the motherboard; adding to it is just as difficult. Think of it as analogous to municipal utilities, such as gas.
and electricity. If you want a different configuration, you’ll have to “move on” to a different motherboard or computer. Incidentally, the term “ROM” is also used, not entirely correctly, when referring to some kinds of storage media that can’t be modified, such as CD-ROMs. Basically, RAM is the size of your playing field, and can be increased as you purchase more “real estate”; ROM is equivalent to your utilities, the hardwired bare necessities needed to operate your computer, and is fixed in size; and storage capacity can be thought of as warehouses of various size, some of them mobile, which can be trucked in or constructed as circumstances warrant.

### 6.4 Principle of working Diode ROM

Diode matrix ROM, used in small amounts in many computers in the 1960s as well as electronic desk calculators and keyboard encoders for terminals. This ROM was programmed by installing discrete semiconductor diodes at selected locations between a matrix of word line traces and bit line traces on a printed circuit board.

Resistor, capacitor, or transformer matrix ROM, used in many computers until the 1970s. Like diode matrix ROM, it was programmed by placing components at selected locations between a matrix of word lines and bit lines. ENIAC’s Function Tables were resistor matrix ROM, programmed by manually setting rotary switches. Various models of the IBM System/360 and complex peripheral devices stored their microcode in either capacitor (called BCROS for balanced capacitor read-only storage on the 360/50 & 360/65, or CCROS for card capacitor read-only Storage on the 360/30) or transformer (called TROS for transformer read-only storage on the 360/20, 360/40 and others) matrix ROM.

Core rope, a form of transformer matrix ROM technology used where size and weight were critical. This was used in NASA/MIT’s Apollo Spacecraft Computers, DEC’s PDP-8 computers, and other places. This type of ROM was programmed by hand by weaving “word line wires” inside or outside of ferrite transformer cores.

Diamond Ring stores, in which wires are threaded through a sequence of large ferrite rings that function only as sensing devices. These were used in TXE telephone exchanges.

The perforated metal character mask (“stencil”) in Chariton cathode ray tubes, which was used as ROM to shape a wide electron beam to form a selected character shape on the Endurance and data retention
6.5 Distinguish between EEPROM and UVROM

6.5.1 EEPROM

Battery EEPROM Works is designed to simplify the process of laptop batteries repair. This process can be divided in two parts: cells replacement and fixing the content of EEPROM or integrated Flash of laptop battery controller. While cells replacement is easy and can be made by any technician who can work with welding machine, EEPROM fixing is very complicated and important task and can be made only by the personal with special skills and equipment.

Battery EEPROM Works makes this process as easy as 1-2-3. The technician needs just to connect EEPROM chip to adapter and press the button. All the necessary work will be made by the software. The laptop battery data will look like a brand new: Full Charge Capacity will be the same as you entered and will reflect real cells capacity, Cycles Count will be set to zero, and Manufacturer Date will be changed to current system date, Permanent Failure Flag will be removed and all the additional necessary changes will be made too. Battery EEPROM Works supports a lot of laptop batteries of different manufactures.

6.5.2 UVROM (Ultraviolet programmable read only memory)

The NVROM is independent from the device. During factory calibration, the host must read the register data from the device and write it to the NVROM. At the start of application, the host must read the register data from the NVROM and write it back to the device, after which the device will wait for further instructions in normal mode.
The serial interface protocol between device and NVPROM is hard coded. A standard NVPROM such as a serial 12C EEPROM with address 0x50 (7-bit) must be used.

### 6.6 List different ROM and RAM IC numbers

- Interfacing with peripheral IC's like 8255.
- Interfacing with peripheral IC's like 8254.
- Interfacing with peripheral IC's like 8279.
- Interfacing with peripheral IC's like 8259.

### 6.7 Working of MOSRAM

**Dynamic random-access memory (DRAM)** is a type of random-access memory that stores each bit of data in a separate capacitor within an integrated circuit. The capacitor can be either charged or discharged; these two states are taken to represent the two values of a bit, conventionally called 0 and 1. Since capacitors leak charge, the information eventually fades unless the capacitor charge is refreshed periodically. Because of this refresh requirement, it is a *dynamic* memory as opposed to SRAM and other *static* memory.

The main memory (the "RAM") in personal computers is dynamic RAM (DRAM). It is the RAM in laptop and workstation computers as well as some of the RAM of video game consoles.

The advantage of DRAM is its structural simplicity: only one transistor and a capacitor are required per bit, compared to four or six transistors in SRAM. This allows DRAM to reach very high densities. Unlike flash memory, DRAM is volatile memory (cf. non-volatile memory), since it loses its data quickly when power is removed. The transistors and capacitors used are extremely small; billions can fit on a single memory chip.

**Operation principle**

Principle of operation of DRAM read, for simple 4 by 4 array.

DRAM is usually arranged in a rectangular array of charge storage cells consisting of one capacitor and transistor per data bit. The figure to the right shows a simple example with a 4 by 4 cell matrix. Modern DRAM matrices are many thousands of cells in height and width.

The long horizontal lines connecting each row are known as word-lines. Each column of cells is composed of two bit-lines, each connected to every...
other storage cell in the column (the illustration to the right does not include this important detail). They are generally known as the + and “ bit-lines.

A sense amplifier is essentially a pair of cross-connected inverters between the bit-lines. The first inverter is connected with input from the + bit-line and output to the “ bit-line. The second inverter’s input is from the “ bit-line with output to the + bit-line. This result in positive feedback which stabilizes after one bit-line is fully at its highest voltage and the other bit-line is at the lowest possible voltage.

A 256 k x 4 bit 20-pin DIP DRAM on an early PC memory card (k = 1024), usually Industry Standard Architecture

Common DRAM packages. From top to bottom: DIP, SIPP, SIMM (30-pin), SIMM (72-pin), DIMM (168-pin), DDR DIMM (184-pin).

### 6.8 Comparisons of Static RAM and Dynamic RAM

**DRAM chip (Integrated Circuit or IC)**

- Dual in-line Package (DIP)
- **DRAM (memory) modules**
  - Single In-line Pin Package (SIPP)
  - Single In-line Memory Module (SIMM)
  - Dual In-line Memory Module (DIMM)
  - Rambus In-line Memory Module (RIMM), technically DIMMs but called RIMMs due to their proprietary slot.
  - Small outline DIMM (SO-DIMM), about half the size of regular DIMMs, are mostly used in notebooks, small footprint PCs (such as Mini-ITX motherboards), upgradable office printers and networking hardware like routers. Comes in versions with:
    - 72-pin (32-bit)
    - 144-pin (64-bit) used for SDRAM
    - 200-pin (72-bit) used for DDR SDRAM and DDR2 SDRAM
    - 204-pin (64-bit) used for DDR3 SDRAM
  - Small outline RIMM (SO-RIMM). Smaller version of the RIMM, used in laptops. Technically SO-DIMMs but called SO-RIMMs due to their proprietary slot.
Stacked vs. non-stacked RAM modules

- Stacked RAM modules contain two or more RAM chips stacked on top of each other. This allows large modules to be manufactured using cheaper low density wafers. Stacked chip modules draw more power, and tend to run hotter than non-stacked modules. Stacked modules can be built using the older TSOP or the newer BGA style IC chips.

Common DRAM modules

Common DRAM packages as illustrated to the right, from top to bottom:

1. DIP 16-pin (DRAM chip, usually pre-fast page mode DRAM (FPRAM))
2. SIPP 30-pin (usually FPRAM)
3. SIMM 30-pin (usually FPRAM)
4. SIMM 72-pin (often extended data out DRAM (EDO DRAM) but FPRAM is not uncommon)
5. DIMM 168-pin (SDRAM)
6. DIMM 184-pin (DDR SDRAM)
7. RIMM 184-pin (RDRAM) not pictured
8. DIMM 240-pin (DDR2 SDRAM and DDR3 SDRAM) not pictured

6.9 Working with NV RAM

The Portfolio operating system treats NVRAM-nonvolatile random access memory-as a file system volume. This maintains consistency for I/O operations; you use the same steps for accessing NVRAM as you do for all other files. Portfolio also provides an NVRAM maintenance utility called lmadm.

A minimally configured 3DO system has at least 32 KB of NVRAM. Application developers can use this facility for persistent storage of small pieces of data. Practical uses include providing a game saving feature, to save user preference settings, or to store application configuration information. There is no way to query the operating system for the amount of available NVRAM memory prior to creating an NVRAM file, because NVRAM is not confined to one contiguous area. The correct procedure is to try to create and allocate the NVRAM file using the Open Disk File call. If the call is successful, then sufficient memory was available. If the call fails for lack of memory, then it returns an error code indicating insufficient space.
When NVRAM is full, some titles cannot manage VRAM appropriately. Some titles fail silently, discarding information the user wants to save. Other titles overwrite information saved by other titles. This can be annoying if a user reached a relatively high level in another title.

**Caution:** Developers should avoid managing NVRAM in the ways mentioned above.

**How to Manage NVRAM**

A title should do the following to manage NVRAM properly:

- Remove obsolete files. For example, files for a previous version of the same title can probably be removed.
- If possible, save files as compressed files.
- Never delete information about a different title without prompting the user.
- Prompt the user to delete information currently in NVRAM if a title attempts to write to NVRAM but fails because it is full. This can be done using Access, discussed in Access: A Simple User Interface. Some guidelines are discussed in the next section.

**Working With NVRAM Files**

Portfolio supports NVRAM access through file system function calls. Your title can use I/O requests to work with an open NVRAM file as a device, as it would any other file. You can use the functions in the section Using NVRAM to create (and allocate) blocks, and to read from, write to, get the block size of a file, and set the end of a file.

This section first provides information on how to choose appropriate filenames, then provides some information about how to look at NVRAM using the 3DO Debugger.

**6.10 Difference between Flash ROM and NVRAM**

*Read-only memory* (ROM) is a class of storage medium used in computers and other electronic devices. Data stored in ROM cannot be modified, or can be modified only slowly or with difficulty, so it is mainly used to distribute firmware.

In its strictest sense, *ROM* refers only to mask ROM (the oldest type of solid state ROM), which is fabricated with the desired data permanently stored in it, and thus can never be modified. Despite the simplicity, speed and economies
of scale of mask ROM, field-programmability often make reprogrammable memories more flexible and inexpensive. As of 2007[update], actual ROM circuitry is therefore mainly used for applications such as microcode, and similar structures, on various kinds of processors.

Other types of non-volatile memory such as erasable programmable read only memory (EPROM) and electrically erasable programmable read-only memory (EEPROM or Flash ROM) are sometimes referred to, in an abbreviated way, as “read-only memory” (ROM); although these types of memory can be erased and re-programmed multiple times, writing to this memory takes longer and may require different procedures than reading the memory. When used in this less precise way, “ROM” indicates a non-volatile memory which serves functions typically provided by mask ROM, such as storage of program code and nonvolatile data.

WORKING: NVRAM are devices which are able to retain information when electrical power is removed.

The first semiconductor NVRAM technology was battery-backed SRAM. It was created by the simple expedient of providing a rechargeable battery to keep power applied to the SRAM when system power was removed. This is still in use and works well for limited time periods, but the batteries take up useful space, and eventually discharge. Computer users who store their computers without power for long periods of time find that the units lose their CMOS setup information because it is typically stored in battery-backed SRAM.

Today, so-called flash memory takes the place of battery-backed SRAM in a number of applications. Most notably, flash has made possible compact “memory sticks,” which are just flash memory chips packaged along with a USB interface. When plugged into a USB port, they appear as a “removable drive.” They serve the same function as floppy disks. Flash memory can be (and is) used for more reliable CMOS Setup storage and virtually any other NVRAM application. Flash’s main drawback is a limitation on the number of read/write cycles its cells can endure.

A second type of NVRAM that is currently gaining popularity is magneto resistive RAM (MRAM). MRAM’s greatest advantage over flash is a virtually unlimited number of read/write cycles.

A third NVRAM technology currently in production is the ferroelectric RAM (FRAM). Like DRAM, FRAM stores information as voltage on a capacitor. Instead of using a linear dielectric, such as silicon dioxide (basically, glass), FRAM uses a non-linear ferroelectric dielectric, such as lead zirconate titanate (PZT). PZT is a crystalline material whose crystal unit cells have a
permanent electric dipole moment. Applying an electric field (by putting a charging voltage across the capacitor) causes atomic rearrangements within the unit cells to align all the dipole moments with the impressed electric field. Removing the supply voltage leaves the dipoles still aligned, so the potential difference between the plates persists.

Other NVRAM technologies under development include phase-change RAM (PRAM), Silicon-Oxide-Nitride-Oxide-Silicon (SONOS), Resistive Random Access Memory (RRAM), Nano-RAM (NRAM), and perhaps others. All of these are under development.

**Short Answer Type Questions**

1. Define read and write operation.
2. Define access time and memory capacity.
3. Define address lines.
4. Define word length or byte.
5. Mention the types of memories.
6. Define cache memory.
7. Mention the types of RAM's.
8. Expand the EEPROM and UVROM.

**Long Answer Type Questions**

1. Explain briefly types memories used in computers.
2. Write comparisons between ROM, RAM.
3. Explain working of diode ROM.
4. Distinguish between EEPROM and UVROM.
5. Explain working of dynamic MOS RAM.
6. Explain briefly memory modules used in computers.

**On Job Training / Practicals Questions**

- Study of different types of RAM, ROM.
- Study of diode RAM working.
- Study of different memory modules used in computers.
Learning Objectives

After studying this unit, the student will be able to

- Study the layout of components, on the motherboard
- Study the expansion slots, chipset specifications
- Study the types of RAMs and cache memory
- Study the AGP, Serial, Parallel and USB ports
- Study of Power supply connectors and external devices
- Study of connector of printer, serial ports, mouse, keyboard
- Study of SMPS and connectors
- Study of SCSI harddisk controllers
- Study of functioning of network card and specifications
- Study of BIOS, operating system and application software
- Study of computer network, router and routing
- Study of working of LAN
- Study of Bluetooth technology
7.0 Computer Hardware and Networking Basics

The Computer mainly consists the functions input, process, output and storage. These functions were described in the manner of diagram as follows.

The Block diagram of computer consists mainly i.e.,

- Input unit
- CPU (Control unit, Main Memory and ALU)
- Output unit,
- Secondary Storage unit

**Fig. 7.1 Block diagram of Computer**

1. **Input:** This is the process of entering data and programs in to the computer system. Therefore, the input unit takes data from us to the computer in an organized manner for processing through an input device such as keyboard, mouse, MICR, OCR, etc.,
2. Main Memory: It is also known as internal memory. It is very fast in operation. It is used to store data and instructions. Data has to be fed into the system before the actual processing starts. It contains a part of the operating system software, one or more execution programs being executed, the data being processed and required by the programs for execution, and processed data awaiting output.

3. Output: This is the process of producing results from the data for getting useful information. Similarly the output produced by the computer after processing must also be kept somewhere inside the computer before being given to you in human readable form through the screen or printer. Again the output is also stored inside the computer for further processing.

4. Control Unit (CU): The next component of computer is the Control Unit, which acts like the supervisor seeing that things are done in proper fashion. Control Unit is responsible for co-ordinating various operations using time signal. The control unit determines the sequence in which computer programs and instructions are executed. Things like processing of programs stored in the main memory, interpretation of the instructions and issuing of signals for other units of the computer to execute them. It also acts as a switch board operator when several users access the computer simultaneously. Thereby it coordinates the activities of computer’s peripheral equipment as they perform the input and output.

5. Arithmetic Logical Unit (ALU): After you enter data through the input device it is stored in the primary storage. The actual processing of the data and instruction are performed by Arithmetic Logical Unit. The major operations performed by the ALU are addition, subtraction, multiplication, division, logic and comparison. Data is transferred to ALU from storage unit when required. After processing the output is returned back to storage unit for further processing or getting stored.

6. Secondary storage: It is also known as auxiliary memory. It is closely linked with the main memory. Since main memory can’t be flooded with unwanted data at particular moment, same is stored in auxiliary memory from which desired data is fed to main memory as and when required by it. Thus secondary storage is used to hold mass of information i.e., system software, application programs, cinemas, games and data files. Obviously the capacity of secondary storage is very high compared to main memory. Auxiliary memory usually in the form of Magnetic disk, Magnetic tape, CD’s, Memory cards, Pen drives etc.
7.1 Layout of Component in the Motherboard

The main printed circuit board in a computer is known as the Motherboard. It is also known as System Board, Main Board or Printed Wired Board (PWB). It also sometimes abbreviated or shortened to mobo. Attached to it, we have numerous motherboard components that are crucial in the functioning of the computer.

The motherboard acts as the connection point where major computer components are attached to. It holds many of the crucial components of the system like the processor, memory, expansion slots and connects directly or indirectly to every part of the PC.

The type of motherboard installed in a PC has a great effect on system speed and expansion capabilities.

![Motherboard Diagram](image)

**Fig. 7.2 Mother Board**

The motherboard holds all the major logic components of the computer. Here we are going to see with no particular order, some of those major motherboard components and their function in a computer or to be more precise in your computer.

![Core 2 Duo Intel Processor](image)

**Fig. 7.3 Core 2 Duo Intel Processor**
Electronics Engineering Technician

CPU- Central Processing Unit

It is also known as the microprocessor or the processor. It is the brain of the computer, and it is responsible for fetching, decoding and executing program instructions as well as performing mathematical and logical calculations.

The processor chip is identified by the processor type and the manufacturer; and this information is usually inscribed on the processor chip e.g. Intel 386, Advanced Micro Devices (AMD) 386, Cyrix 486, Pentium MMX, (old processor types) Intel Core 2Duo e.t.c.

If the processor chip is not on the motherboard, you can identify the processor socket as socket 1 to Socket 8, LGA 775 e.t.c. This can help you identify the processor that fits in the socket. For example a 486DX processor fits into Socket 3.

Main Memory (RAM)

Random access memory or RAM most commonly refers to computer chips that temporarily store dynamic data when you are working with your computer to enhance the computer performance. In other words, it is the working place of your computer where active programs and data are loaded so that any time the processor requires them, it doesn’t have to fetch them from the hard disk which will take a longer access time.

![Fig. 7.3 RAM](image)

Random access memory is volatile memory, meaning it loses its contents once power is turned off. This is different from non-volatile memory such as hard disks and flash memory, which do not require a power source to retain data. When a computer shuts down properly, all data located in random access memory is returned back to permanent storage on the hard drive or flash drive. At the next boot-up, RAM begins to fill with programs automatically loaded at startup, and with files opened by the user a process called booting.

BIOS - Basic Input Output System

BIOS is a term that stands for basic input/output system, which consists of low-level software that controls the system hardware and acts as an interface
between the operating system and the hardware. Most people know the term BIOS by another name, device drivers, or just drivers. In other words, the BIOS is drivers, meaning all of them. BIOS is essentially the link between hardware and software in a system.

All motherboards include a small block of Read Only Memory (ROM) which is separate from the main system memory used for loading and running software. On PCs, the BIOS contains all the code required to control the keyboard, display screen, disk drives, serial communications, and a number of miscellaneous functions.

The system BIOS is a ROM chip on the motherboard used by the computer during the startup routine (boot process) to check out the system and prepare to run the hardware. The BIOS is stored on a ROM chip because ROM retains information even when no power is being supplied to the computer.

![Fig. 7.3 BIOS Chip](image)

**CMOS-Complimentary Metal Oxide Semiconductor**

Motherboards also include a small separate block of memory made from CMOS RAM chips which is kept alive by a battery (known as a CMOS battery) even when the PC’s power is off. This prevents reconfiguration when the PC is powered on.

CMOS devices require very little power to operate.

The CMOS RAM is used to store basic information about the PC’s configuration e.g.

- Floppy disk and hard disk drive types
- CPU
- RAM size
- Date and time
· Serial and parallel port information
· Plug and Play information
· Power Saving settings

The Other Important data kept in CMOS memory is the time and date, which is updated by a Real Time Clock (RTC).

![CMOS Battery](image)

**Fig. 7.4 CMOS Battery**

**Cache Memory**

It is a small block of high-speed memory (RAM) that enhances PC performance by pre-loading information from the (relatively slow) main memory and passing it to the processor on demand.

Most CPUs have an internal cache (in-built in the processor) which is referred to as Level-I cache memory or primary cache memory. This can be supplemented by external cache memory fitted on the motherboard. This is the Level-2 Cache memory or secondary cache. Some CPUs have both L1 and L2 cache built-in and designate the separate cache chip as Level 3 (L3) cache.

![Cache Memory](image)

**Fig. 7.5 Cache Memory**
Expansion Buses

An input/output pathway from the CPU to peripheral devices typically made up of a series of slots on the motherboard. Expansion boards (cards) plug into the bus. PCI is the common expansion bus in a PC and other hardware platforms. Buses carry signals, such as data; memory addresses, power and control signals from component to component.

Expansion buses enhance the PCs capabilities by allowing users to add missing features in their computers in form of adapter cards that are slotted in expansion slots.

Fig. 7.6 PCI Slots

Chip Sets

A chipset is a group of small circuits that coordinate the flow of data to and from key components of a PC. This includes the CPU itself, the main memory, the secondary cache and any devices situated on the buses. The chipset also controls data flow to and from hard disks, and other devices connected to the IDE channels. A computer has got two main chipsets:-

- The NorthBridge (also called the memory controller) is in charge of controlling transfers between the processor and the RAM, which is way it is located physically near the processor. It is sometimes called the GMCH, for Graphic and Memory Controller Hub.

- The SouthBridge (also called the input/output controller or expansion controller) handles communications between peripheral devices. It is also called the ICH (I/O Controller Hub). The term bridge is generally used to designate a component which connects two buses.

Chipset manufacturers include SIS, VIA, ALI, OPTI e.t.c
Cpu Clock

The clock synchronizes the operation of all parts of the PC and provides the basic timing signal for the CPU. Using a quartz crystal, the CPU clock breathes life into the microprocessor by feeding it a constant flow of pulses. For example, a 200 MHz CPU receives 200 million pulses per second from the clock. A 2 GHz CPU gets two billion pulses per second. Similarly, in a communications device, a clock may be used to synchronize the data pulses between sender and receiver.

A “real-time clock,” also called the “system clock,” keeps track of the time of day and makes this data available to the software. A “timesharing clock” interrupts the CPU at regular intervals and allows the operating system to divide its time between active users and/or applications.

Switches and Jumpers

DIP (Dual In-line Package) switches are small electronic switches found on the circuit board that can be turned on or off just like a normal switch. They are very small and so are usually flipped with a pointed object such as a screwdriver, bent paper clip or pen top. Care should be taken when cleaning near DIP switches as some solvents may destroy them.

Jumper Pins

Jumpers are small pins on the board with plastic or metal devices that go over the pins. This device is called a bridge or a jumper cap. When the bridge is connected to any two pins via a shorting link, it completes the circuit and a certain configuration has been achieved.
Jumper cap

A metal bridge that closes an electrical circuit. Typically, a jumper consists of a plastic plug that fits over a pair of protruding pins. Jumpers are sometimes used to configure expansion boards. By placing a jumper plug over a different set of pins, you can change a board’s parameters.

7.2 Expansion Slots on Mother Board

There are many different types of motherboard expansion slots, but they all have one thing in common: They allow you to plug expansion cards into your computer and increase its functionality. While motherboards often come with on-board sound, wired networking, and video, plugging in a dedicated expansion card will result in this card being used instead of the motherboard’s built-in hardware and can increase your computer’s gaming, video playback, or sound performance. Expansion cards can also provide new functionality, such as allowing your computer to capture TV signals or access a wireless network.

Read on for information about common expansion slot types and functionality.

PCI Express

PCI Express (or PCIe) is the newest standard for expansion cards on personal computers. PCI Express is meant to replace older standards like PCI and AGP, mentioned below. PCIe provides significantly more bandwidth, allowing for higher performance video cards and network cards. Video cards in particular are the most common consumer use of these slots, since they need high bandwidth for maximum 3D gaming and graphics performance.

While PCI Express is meant to replace the AGP and PCI standards, many PCI cards are still being manufactured, particularly for expansion cards which do not need the increase in bandwidth provided.
PCI Express is now dominating however, and motherboards are being manufactured with fewer PCI slots and more PCIe slots.

PCI (Peripheral Component Interconnect) is not to be confused with PCI Express, which is meant to replace it. Unlike PCI Express, PCI is an older standard which provides less bandwidth for expansion cards. In spite of the fact that the standard was created in 1993, new motherboards still ship with PCI slots for compatibility purposes.

PCI cards are still very common for expansion cards that do not need high bandwidth, such as most sound cards, network cards, USB expansion cards for additional connections, and more. Since newer motherboards still tend to come with PCI slots for compatibility, PCI cards will function on most computers. In contrast, PCI Express cards will only function on newer computers. Manufacturers can release expansion cards which function with most computers if the cards are PCI.

The AGP (Accelerated Graphics Port) expansion slot standard was introduced when video cards needed more bandwidth for performance than was provided by PCI. As suggested by the title, AGP slots are used for video cards. However, AGP has been largely phased out in favour of the PCI Express expansion slot standard. Unlike AGP, PCI Express provides higher bandwidth for other types of expansion cards that could use it, such as some newer, high-performance sound and network cards.

ExpressCard & PC Card (or PCMCIA)

These standards are designed to be used with portable computers such as laptops. ExpressCard is the successor to PC Card (also known as PCMCIA), and, like PCIe over PCI, has more bandwidth.
Unlike PCI, these types of cards are hot-pluggable (which means you can plug them in while your laptop is running, without shutting it down first).

### 7.3 Functions of Chipset & Chip set Numbers in use

The chipset normally consists of two major microchips. These are known as the North bridge and the South Bridge. Developments in chip technologies have meant that chipset and CPU manufacturers are changing the way the chipset layout works, for example some CPU’s come with a built in memory controller taking that job from the North Bridge, some chipsets have incorporated the north and south bridge in the same chip, but for now we will look at the standard setup.

**Chip Set Numbers in use**

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8085</td>
<td>Core Duo</td>
</tr>
<tr>
<td>8086</td>
<td>Core i3</td>
</tr>
<tr>
<td>8087</td>
<td>Core i3 Mobile</td>
</tr>
<tr>
<td>8088</td>
<td>Core i5</td>
</tr>
<tr>
<td>80186</td>
<td>Core i5 Mobile</td>
</tr>
<tr>
<td>80286</td>
<td>Core i7</td>
</tr>
<tr>
<td>80376</td>
<td>Core i7 Mobile</td>
</tr>
<tr>
<td>80386</td>
<td>Core i9</td>
</tr>
<tr>
<td>80387</td>
<td>Core Solo</td>
</tr>
<tr>
<td>80486</td>
<td>i186</td>
</tr>
<tr>
<td>80486 overdrive</td>
<td>i286</td>
</tr>
<tr>
<td>80487</td>
<td>i386</td>
</tr>
<tr>
<td>80860</td>
<td>i387</td>
</tr>
<tr>
<td>80960</td>
<td>i486</td>
</tr>
<tr>
<td>Atom</td>
<td>i487</td>
</tr>
<tr>
<td>Celeron</td>
<td>i860</td>
</tr>
<tr>
<td>Celeron (Mobile)</td>
<td>i960</td>
</tr>
<tr>
<td>Celeron D</td>
<td>Itanium</td>
</tr>
<tr>
<td>Celeron Dual-Core</td>
<td>Itanium 2</td>
</tr>
<tr>
<td>Celeron Dual-Core (Mobile)</td>
<td>Mobile Celeron</td>
</tr>
<tr>
<td>Celeron M</td>
<td>Mobile Celeron Dual-Core</td>
</tr>
<tr>
<td>Core 2 Duo (Desktop)</td>
<td>Mobile Pentium 4</td>
</tr>
<tr>
<td>Core 2 Duo (Mobile)</td>
<td>Mobile Pentium 4-M</td>
</tr>
<tr>
<td>Core 2 Solo</td>
<td>Mobile Pentium II</td>
</tr>
<tr>
<td>Mobile Pentium III</td>
<td>Mobile Pentium III-M</td>
</tr>
<tr>
<td>Other chips (not processors)</td>
<td>Pentium</td>
</tr>
<tr>
<td>Pentium 4</td>
<td>Pentium 4 (Mobile)</td>
</tr>
<tr>
<td>Pentium 4-M (Mobile)</td>
<td>Pentium D</td>
</tr>
<tr>
<td>Pentium Dual-Core</td>
<td>Pentium Dual-Core (Mobile)</td>
</tr>
<tr>
<td>Pentium Extreme Edition</td>
<td>Pentium II</td>
</tr>
<tr>
<td>Pentium II</td>
<td>Pentium II (Mobile)</td>
</tr>
<tr>
<td>Pentium II Overdrive</td>
<td>Pentium II Xeon</td>
</tr>
<tr>
<td>Pentium III</td>
<td>Pentium III (Mobile)</td>
</tr>
<tr>
<td>Pentium III Xeon</td>
<td>Pentium III-M (Mobile)</td>
</tr>
<tr>
<td>Pentium M</td>
<td>Pentium Overdrive</td>
</tr>
<tr>
<td>Pentium Pro</td>
<td>Pentium Pro</td>
</tr>
<tr>
<td>Timna</td>
<td>Xeon</td>
</tr>
</tbody>
</table>
The North Bridge handles data for the graphics port whether that be AGP or PCI express and the main memory which includes the FSB (Front side bus). Although both chips are required for the PC to work the North Bridge handles most of the very important tasks such as the connection between the CPU and main memory bank. The South Bridge handles data from the PCI x1 slots and can also have integrated components such as Audio and/or onboard graphics.

The North and South bridges will have different chip names even though they are very often paired with the same opposite bridge to come under the collective name of the chipset.

### 7.4 Specifications of Processor

<table>
<thead>
<tr>
<th>Status</th>
<th>Launched</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launch Date</td>
<td>Q3’08</td>
</tr>
<tr>
<td>Processor Number</td>
<td>E8600</td>
</tr>
<tr>
<td># of Cores</td>
<td>2</td>
</tr>
<tr>
<td># of Threads</td>
<td>2</td>
</tr>
<tr>
<td>Clock Speed</td>
<td>3.33 GHz</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>6 MB</td>
</tr>
<tr>
<td>Bus/Core Ratio</td>
<td>10</td>
</tr>
<tr>
<td>FSB Speed</td>
<td>1333 MHz</td>
</tr>
<tr>
<td>FSB Parity</td>
<td>No</td>
</tr>
<tr>
<td>Instruction Set</td>
<td>64-bit</td>
</tr>
<tr>
<td>Embedded Options Available</td>
<td>No</td>
</tr>
<tr>
<td>Supplemental SKU</td>
<td>No</td>
</tr>
<tr>
<td>Lithography</td>
<td>45 nm</td>
</tr>
<tr>
<td>Max TDP</td>
<td>65 W</td>
</tr>
<tr>
<td>VID Voltage Range</td>
<td>0.8500V-1.3625V</td>
</tr>
<tr>
<td>Recommended Customer Price</td>
<td>TRAY: $266 BOX: $285</td>
</tr>
</tbody>
</table>

### Package Specifications

| T<sub>CASE</sub>     | 72.4°C           |
| Package Size         | 37.5mm x 37.5mm  |
| Processing Die Size  | 107 mm<sup>2</sup> |
| # of Processing Die Transistors | 410 million |
| Sockets Supported    | LGA775           |
| Low Halogen Options Available | See MDDS |
| Intel® Turbo Boost Technology | No |
| Intel® Hyper-Threading Technology | No |
| Intel® Virtualization Technology (VT-x) | Yes |
| Intel® Virtualization Technology for Directed I/O (VT-d) | Yes |
| Intel® Trusted Execution Technology | Yes |
| AES New Instructions | Yes             |
| Intel® 64            | No               |
| Idle States          | Yes              |
| Enhanced Intel SpeedStep® Techno. | Yes |
7.5 Different Types of RAM in use

Types of RAM

RAM (Random Access Memory) is a temporary (Volatile) storage area utilized by the CPU. Before a program can be ran the program is loaded into the memory which allows the CPU direct access to the program.

1. SRAM
2. DRAM
3. VRAM
4. NVRAM
5. MOS RAM

SRAM Short for static random access memory, and pronounced ess-ram. SRAM is a type of memory that is faster and more reliable than the more common DRAM (dynamic RAM). The term static is derived from the fact that it doesn’t need to be refreshed like dynamic RAM.

RAM is often used only as a memory cache usually found in the CPU (L1, L2 and L3 Cache)

DRAM stands for dynamic random access memory, a type of memory used in most personal computers.

Types of DRAM Packages and DRAM Memory

Fig. 7.11 (72, 144, 200) SO-DIMM

SO-DIMM Short for Small Outline DIMM, a small version of a DIMM used commonly in notebook computers. 72 supports 32bit and 144 and 200 SO-DIMM pins support a full 64-bit transfer.

Micro-DIMM Short for Micro Dual Inline Memory Module, a competing memory used on laptops, mostly supports 144 and 172 pins. Micro-DIMM Short for Micro Dual Inline Memory Module, a competing memory used on laptops, mostly supports 144 and 172 pins.
SIMM  Acronym for single in-line memory module, a small circuit board that can hold a group of memory chips. Typically, SIMM’s holds up 8 (on Macintoshes) or 9 (on PCs) RAM chips. On PCs, the ninth chip is often used for parity error checking. Unlike memory chips, SIMM’s is measured in bytes rather than bits. SIMM’s is easier to install than individual memory chips. A SIMM is either 30 or 72 pins.

FPM RAM  Short for Fast Page Mode RAM, a type of Dynamic RAM (DRAM) that allows faster access to data in the same row or page. Page-mode memory works by eliminating the need for a row address if data is located in the row previously accessed. It is sometimes called page mode memory.

EDO DRAM  Short for Extended Data Output Dynamic Random Access Memory, a type of DRAM that is faster than conventional DRAM. Unlike conventional DRAM which can only access one block of data at a time, EDO RAM can start fetching the next block of memory at the same time that it sends the previous block to the CPU.

DIMM  Short for dual in-line memory module, a small circuit board that holds memory chips. A single in-line memory module (SIMM) has a 32-bit path to the memory chips whereas a DIMM has a 64-bit path. Because the Pentium
processor requires a 64-bit path to memory, you need to install SIMM’s two at a time. With DIMM’s, you can install memory one DIMM at a time. A DIMM contains 168 pins.

**Fig. 7.15  168 pin DIMM (SDRAM)**

**SDRAM** Short for Synchronous DRAM, a new type of DRAM that can run at much higher clock speeds than conventional memory. SDRAM actually synchronizes itself with the CPU’s bus and is capable of running at 133 MHz, about three times faster than conventional FPM RAM, and about twice as fast EDO DRAM. SDRAM is replacing EDO DRAM in many newer computers.

SDRAM delivers data in high speed burst

**Fig. 7.16  184 pin DIMM (DDR-SDRAM)**

**DDR SDRAM** Short for Double Data Rate-Synchronous DRAM, a type of SDRAM that supports data transfers on both edges of each clock cycle, effectively doubling the memory chip’s data throughput. DDR-SDRAM is also called SDRAM II.

**Fig. 7.17  240 DIMM (DDR2-SDRAM)**

**DDR2-SDRAM** Short for Double Data Rate Synchronous DRAM 2 is a type of DDR that supports higher’s speeds than it’s predecessor DDR SDRAM.
7.6 Cache Memory

Cache memory is random access memory (RAM) that a computer microprocessor can access more quickly than it can access regular RAM. As the microprocessor processes data, it looks first in the cache memory and if it finds the data there (from a previous reading of data), it does not have to do the more time-consuming reading of data from larger memory.

Cache memory is sometimes described in levels of closeness and accessibility to the microprocessor. An L1 cache is on the same chip as the microprocessor. (For example, the PowerPC 601 processor has a 32 kilobyte level-1 cache built into its chip.) L2 is usually a separate static RAM (SRAM) chip. The main RAM is usually a dynamic RAM (DRAM) chip.

In addition to cache memory, one can think of RAM itself as a cache of memory for hard disk storage since all of RAM’s contents come from the hard disk initially when you turn your computer on and load the operating system (you are loading it into RAM) and later as you start new applications and access new data. RAM can also contain a special area called a disk cache that contains the data most recently read in from the hard disk.

7.7 Accelerated Graphics Port / Card

Short for Accelerated Graphics Port, AGP is an advanced port designed for Video cards and 3D accelerators. Designed by Intel and introduced in August of 1997, AGP introduces a dedicated point-to-point channel that allows the graphics controller direct access the system memory. Below is an illustration of what the AGP slot may look like on your motherboard.
The AGP channel is 32-bits wide and runs at 66 MHz. This translates into a total bandwidth of 266 MBps, which is much greater than the PCI bandwidth of up to 133 MBps. AGP also supports two optional faster modes, with throughput of 533 MBps and 1.07 GBps. It also allows 3-D textures to be stored in main memory rather than video memory.

Each computer with AGP support will either have one AGP slot or onboard AGP video. If you needed more than one video card in the computer, you can have one AGP video card and one PCI video card or use a motherboard that supports SLI.

7.8 Power Supply Connectors and External Devices

A power supply unit (PSU) converts mains AC to low-voltage regulated DC power for the internal components of a computer. Modern personal computers universally use a switched-mode power supply. Some power supplies have a manual selector for input voltage, while others automatically adapt to the supply voltage.

Most modern desktop personal computer power supplies conform to the ATX form factor. ATX power supplies are turned on and off by a signal from the motherboard. They also provide a signal to the motherboard to indicate when the DC power lines are correct so that the computer is able to boot up. While an ATX power supply is connected to the mains supply it provides a 5 V standby (5VSB) line so that the standby functions on the computer and certain peripherals are powered. The most recent ATX PSU standard is version 2.31 of mid-2011.
External Connections

+ 5V is given to mother board.

+ 12 V is given to HDD

- 12 Motors of HDD and FDD

- 5V is given to other peripherals

+ 12 V is given to the printer

+ 12 V is given to Scanner

7.9 Serial, Parallel, and USB Ports

Serial Port

An Asynchronous port on the computer used to connect a serial device to the computer and capable of transmitting one bit at a time. Serial ports are typically identified on IBM compatible computers as COM (communications) ports. For example, a mouse might be connected to COM1 and a modem to COM2. With the introduction of USB, FireWire, and other faster solutions serial ports are rarely used when compared to how often they’ve been used in the past. In the picture to the right is a close up of a DB9 serial port on the back of a computer.

Below is a listing of various hardware components that can be purchased and used with your serial port.

Mouse - One of the most commonly used devices for serial ports, usually used with computers with no PS/2 or USB ports and specialty mice.

Modem - Another commonly used device for serial ports. Used commonly with older computers, however, is also commonly used for its ease of use.

Network - One of the original uses of the serial port, which allowed two computers to connect together and allow large files to be transferred between the two.

Printer - Today, this not a commonly used device for serial ports. However, was frequently used with older printers and plotters.
Parallel Port

A **parallel port** is a type of interface found on computers (personal and otherwise) for connecting various peripherals. In computing, a parallel port is a parallel communication physical interface. It is also known as a **printer port** or **Centronics port**. The IEEE 1284 standard defines the bi-directional version of the port, which allows the transmission and reception of data bits at the same time.

**USB Ports**

A USB port is a standard cable connection interface on personal computers and consumer electronics. USB ports allow stand-alone electronic devices to be connected via cables to a computer (or to each other).

USB stands for Universal Serial Bus, an industry standard for short-distance digital data communications. USB allows data to be transferred between devices. USB ports can also supply electric power across the cable to devices without their own power source.

Both wired and wireless versions of the USB standard exist, although only the wired version involves USB ports and cables.

Many types of consumer electronics support USB interfaces. These types of equipment are most commonly used for computer networking:

- USB network adapters
- USB broadband and cellular modems for Internet access
- USB printers to be shared on a home network

For computer-to-computer file transfers without a network, USB keys are also sometimes used to copy files between devices.

Multiple USB devices can also be connected to each other using a USB hub. A USB hub plugs into one USB port and contains additional ports for other devices to connect subsequently.

### 7.10 Connector Details for Printer, Serial Port, Mouse, Keyboard and USB

#### Keyboard Connector

<table>
<thead>
<tr>
<th>Connector Pin #</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>KBDCLK (clock)</td>
</tr>
</tbody>
</table>
Electronics Engineering Technician

Pin 2 KBDAT (data)
Pin 3 KBRST (reset, not used)
Pin 4 GND
Pin 5 VCC (+5V)

Fig. 7.21 Keyboard Connector

Mouse Connector

<table>
<thead>
<tr>
<th>Connector Pin #</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>KBDAT (data)</td>
</tr>
<tr>
<td>Pin 2</td>
<td>not used</td>
</tr>
<tr>
<td>Pin 3</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 4</td>
<td>VCC (+5V)</td>
</tr>
<tr>
<td>Pin 5</td>
<td>KBDCLK (clock)</td>
</tr>
<tr>
<td>Pin 6</td>
<td>not used</td>
</tr>
</tbody>
</table>

Fig. 7.22 Mouse Connector
Printer Parallel Port

25 pin Parallel Port Connector

25 pin “D” connector

Connector may be reversed depending on which side is viewed. All pins are numbered.

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Function</th>
<th>Pin No.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Strobe</td>
<td>14</td>
<td>Auto Feed</td>
</tr>
<tr>
<td>2</td>
<td>Data 0</td>
<td>15</td>
<td>Error</td>
</tr>
<tr>
<td>3</td>
<td>Data 1</td>
<td>16</td>
<td>Init</td>
</tr>
<tr>
<td>4</td>
<td>Data 2</td>
<td>17</td>
<td>Select In</td>
</tr>
<tr>
<td>5</td>
<td>Data 3</td>
<td>18</td>
<td>Ground</td>
</tr>
<tr>
<td>6</td>
<td>Data 4</td>
<td>19</td>
<td>Ground</td>
</tr>
<tr>
<td>7</td>
<td>Data 5</td>
<td>20</td>
<td>Ground</td>
</tr>
<tr>
<td>8</td>
<td>Data 6</td>
<td>21</td>
<td>Ground</td>
</tr>
<tr>
<td>9</td>
<td>Data 7</td>
<td>22</td>
<td>Ground</td>
</tr>
<tr>
<td>10</td>
<td>Acknowledge</td>
<td>23</td>
<td>Ground</td>
</tr>
<tr>
<td>11</td>
<td>Busy</td>
<td>24</td>
<td>Ground</td>
</tr>
<tr>
<td>12</td>
<td>Paper Empty</td>
<td>25</td>
<td>Ground</td>
</tr>
<tr>
<td>13</td>
<td>Select</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Mouse

IBM PS/2-type mouse and keyboard connectors are both “6-pin mini-DIN” connectors. The female connectors are on the computer, the male connectors are on the mouse and keyboard cables that plug into the computer.
Viewing the female connector (on the computer), the pinout is:

![Male Mouse Connector Diagram]

**Fig. 7.24 Male Mouse Connector**

Viewing the male connector (on the keyboard or mouse cable), the pinout is: In both cases, the pins are normally used as follows:

**Pin Function**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data</td>
</tr>
<tr>
<td>2</td>
<td>(Reserved)</td>
</tr>
<tr>
<td>3</td>
<td>Gnd</td>
</tr>
<tr>
<td>4</td>
<td>+5v</td>
</tr>
<tr>
<td>5</td>
<td>Clock</td>
</tr>
<tr>
<td>6</td>
<td>(Reserved)</td>
</tr>
</tbody>
</table>

### 7.11 SMPS Voltages Connector Detail

SMPS supplies power to all components of a PC located in the CPU viz. motherboard, hard disk, floppy drive, CD ROM, fans etc. Most computer SMPS’s have the appearance of a square metal box. They have a fan installed at one end and a large bundle of wires emerge out from the other end having:

- One PC Main connector (20 or 24 pins) for power supply to the motherboard.
- 4 pin Peripheral connectors for power supply to Hard Disk Drive, CD/DVD Drive etc.
- One 4 pin floppy connector for power supply to floppy drive.
- One Serial ATA connector for power supply to SATA devices.

Modern desktop SMPS’s produce DC voltages of +/-5V, +3.3V and +/-12V outputs. The DC-DC converters on the motherboard step down SMPS
voltage to the CPU core voltage and other low voltages needed for other motherboard components.

### 7.12 SCSI Hard Disk Controller

The disk controller (or “hard disk controller”) is the circuit which allows the CPU to communicate with a hard disk, floppy disk or other kind of disk drive. The most common are IDE and SCSI controllers. Most home personal computers use IDE controllers, whereas, high end PCs, professional workstations and network file servers generally have SCSI adaptors.

SCSI (Small Computer System Interface) was originally a generic interface for data exchanges between an array of devices. SCSI is regularly used in high-performance servers and workstations. VirtualBox supports LsiLogic and BusLogic SCSI controllers, due to compatibility reasons. As many as 16 virtual hard disks can be attached to each.

### 7.13 Functions of Network Card and Specifications

A network adapter card links a PC with the network cabling system. The network adapter card fits into one of the expansion slots of the PC. The card has one or more user-accessible ports to which the network cabling medium is connected.

Like other hardware devices, a network adapter card has a driver, a software component that manages the device. The network adapter card driver serves a crucial role in the networking architecture. Adapter card drivers inhabit the Data Link layer of the OSI model or, more specifically, the Media Access Control (MAC) sub layer of the Data Link layer. A network adapter card driver is sometimes called a MAC driver. The network adapter card’s built-in ROM (read only memory) performs the functions of the Logical Link Control sub layer of the OSI Data Link layer.

![Fig. 7.25 Network Card and Slot](image)
The network adapter card and its accompanying software and firmware perform several roles. Microsoft identifies the following roles for the network adapter card:

- Preparing data for the transmission medium
- Sending data
- Controlling the flow of data

Preparing Data for the Transmission Medium

The data travels on the network in serial form (one bit at a time). Inside the PC, however, data moves along the data bus (The data bus is a pathway inside your computer that carries data between the hardware components) in parallel form (8, 16, or 32 bits at a time). The network adapter card, therefore, must convert the parallel data from the bus into the serial form required for network transmission.

Sending Data

The network adapter card places data on the network and receives data from the network. These tasks require a certain amount of data flow control. The network adapter card receives packets from the network, checks the destination address of all packets and then interrupts the CPU only if the packet is addressed to the local system.

Controlling the Flow of Data

For two computers to exchange data, the computers’ network adapter cards must be in agreement on certain transmission parameters. A newer card with a higher maximum transmission rate, for instance, might have the capability to use a data with lower transmission rate in order to communicate with a slower card. Before sending data, the cards exchange messages and agree upon such parameters as a transmission speed and a time interval between packets.

Specification of Network card

The Network Driver Interface Specification describes an interface by which one or more NIC drivers communicate with one or more overlying protocol drivers and the operating system. NDIS provides a full abstraction for the network driver development. The NIC drivers can rely upon the NDIS for all the external functions including communication with protocol drivers, registering and intercepting NIC hardware interrupts and communicating with underlying NICs.
The Basic Input Output System (BIOS), also known as the System BIOS or ROM, is a de facto standard defining a firmware interface. The name originated as the name of a component of CP/M (circa 1973-1974), where the BIOS was loaded from disc rather than stored as firmware on ROM (because ROMs were expensive and difficult to reprogram at the time).

The BIOS software is built into the PC, and is the first code run by a PC when powered on (‘boot firmware’). When the PC starts up, the first job for the BIOS is the power-on self-test, which initializes and identifies system devices such as the CPU, RAM, video display card, keyboard and mouse, hard disk drive, optical disc drive and other hardware. The BIOS then locates boot loader software held on a peripheral device (designated as a ‘boot device’), such as a hard disk or a CD/DVD, and loads and executes that software, giving it control of the PC. This process is known as booting, or booting up, which is short for bootstrapping.

BIOS software is stored on a non-volatile ROM chip on the motherboard. It is specifically designed to work with each particular model of computer, interfacing with various devices that make up the complementary chipset of the system. In modern computer systems the BIOS chip’s contents can be rewritten without removing it from the motherboard, allowing BIOS software to be upgraded in place.

Operating System

The 1960's definition of an operating system is “the software that controls the hardware”. However, today, due to microcode we need a better definition. We see an operating system as the programs that make the hardware useable. In brief, an operating system is the set of programs that controls a computer. Some examples of operating systems are UNIX, Mach, MS-DOS, MS-Windows, Windows/NT, Chicago, OS/2, MacOS, VMS, MVS, and VM.

Controlling the computer involves software at several levels. We will differentiate kernel services, library services, and application-level services, all of which are part of the operating system. Processes run Applications, which are linked together with libraries that perform standard services. The kernel supports the processes by providing a path to the peripheral devices. The kernel responds to service calls from the processes and interrupts from the devices. The core of the operating system is the kernel, a control program that functions in privileged state (an execution context that allows all hardware instructions
to be executed), reacting to interrupts from external devices and to service requests and traps from processes. Generally, the kernel is a permanent resident of the computer. It creates and terminates processes and responds to their request for service.

Operating Systems are resource managers. The main resource is computer hardware in the form of processors, storage, input/output devices, communication devices, and data. Some of the operating system functions are: implementing the user interface, sharing hardware among users, allowing users to share data among themselves, preventing users from interfering with one another, scheduling resources among users, facilitating input/output, recovering from errors, accounting for resource usage, facilitating parallel operations, organizing data for secure and rapid access, and handling network communications.

**Objectives of Operating Systems**

Modern Operating systems generally have following three major goals. Operating systems generally accomplish these goals by running processes in low privilege and providing service calls that invoke the operating system kernel in high-privilege state.

**Application Software**

**Application software**, also known as an **application** or an **app**, is computer software designed to help the user to perform specific tasks. Examples include enterprise software, accounting software, office suites, graphics software and media players. Many application programs deal principally with documents. Apps may be bundled with the computer and its system software, or may be published separately. Some users are satisfied with the bundled apps and need never install one.

Application software is contrasted with system software and middleware, which manage and integrate a computer’s capabilities, but typically do not directly apply in the performance of tasks that benefit the user. The system software serves the application, which in turn serves the user.

Similar relationships apply in other fields. For example, a shopping mall does not provide the merchandise a shopper is seeking, but provides space and services for retailers that serve the shopper. A bridge may similarly support rail tracks which support trains, allowing the trains to transport passengers.

Application software applies the power of a particular computing platform or system software to a particular purpose. Some applications are available in versions for several different platforms; others have narrower requirements and
are thus called, for example, a Geography application for Windows or an Android application for education or Linux gaming. Sometimes a new and popular application arises which only runs on one platform, increasing the desirability of that platform. This is called a killer application.

7.15 Appreciation of Networking of Computers

A network is a system that provides its users with unique capabilities, above and beyond what the individual machines and their software applications can provide.

Most of the benefits of networking can be divided into two generic categories: connectivity and sharing. Networks allow computers, and hence their users, to be connected together. They also allow for the easy sharing of information and resources, and cooperation between the devices in other ways. Since modern business depends so much on the intelligent flow and management of information, this tells you a lot about why networking is so valuable.

Here, in no particular order, are some of the specific advantages generally associated with networking:

**Connectivity and Communication:** Networks connect computers and the users of those computers. Individuals within a building or work group can be connected into local area networks (LANs); LANs in distant locations can be interconnected into larger wide area networks (WANs). Once connected, it is possible for network users to communicate with each other using technologies such as electronic mail. This makes the transmission of business (or non-business) information easier, more efficient and less expensive than it would be without the network.

**Data Sharing:** One of the most important uses of networking is to allow the sharing of data. Before networking was common, an accounting employee who wanted to prepare a report for her manager would have to produce it on his PC, put it on a floppy disk, and then walk it over to the manager, who would transfer the data to her PC’s hard disk. (This sort of “shoe-based network” was sometimes sarcastically called a “sneakernet”.)

True networking allows thousands of employees to share data much more easily and quickly than this. More so, it makes possible applications that rely on the ability of many people to access and share the same data, such as databases, group software development, and much more. Intranets and extranets can be used to distribute corporate information between sites and to business partners.
Hardware Sharing: Networks facilitate the sharing of hardware devices. For example, instead of giving each of 10 employees in a department an expensive color printer (or resorting to the “sneakernet” again), one printer can be placed on the network for everyone to share.

Internet Access: The Internet is itself an enormous network, so whenever you access the Internet, you are using a network. The significance of the Internet on modern society is hard to exaggerate, especially for those of us in technical fields.

Internet Access Sharing: Small computer networks allow multiple users to share a single Internet connection. Special hardware devices allow the bandwidth of the connection to be easily allocated to various individuals as they need it, and permit an organization to purchase one high-speed connection instead of many slower ones.

Data Security and Management: In a business environment, a network allows the administrators to much better manage the company’s critical data. Instead of having this data spread over dozens or even hundreds of small computers in a haphazard fashion as their users create it, data can be centralized on shared servers. This makes it easy for everyone to find the data, makes it possible for the administrators to ensure that the data is regularly backed up, and also allows for the implementation of security measures to control who can read or change various pieces of critical information.

Performance Enhancement and Balancing: Under some circumstances, a network can be used to enhance the overall performance of some applications by distributing the computation tasks to various computers on the network.

Entertainment: Networks facilitate many types of games and entertainment. The Internet itself offers many sources of entertainment, of course. In addition, many multi-player games exist that operate over a local area network. Many home networks are set up for this reason, and gaming across wide area networks (including the Internet) has also become quite popular. Of course, if you are running a business and have easily-amused employees, you might insist that this is really a disadvantage of networking and not an advantage.

7.16 Concept of Router and Routing

Large amounts of bandwidth can be provided easily and relatively inexpensively in a local area network (LAN). However, providing high bandwidth between a local network and the Internet can be very expensive. Because of this expense, Internet access is usually provided by a slower-speed wide-area network (WAN) link such as a cable or DSL modem. For the WAN link to
work on the Internet, the data traffic meant for the Internet needs to be separated from other WAN data and forwarded. A router usually performs the tasks of selecting and forwarding this data.

Routing is the process of transferring data across an internetwork from a source host to a destination host. Routing can be understood in terms of two processes: host routing and router routing.

Host routing occurs when the sending host forwards a packet. Based on the destination network address, the sending host must decide whether to forward the packet to the destination or to a router. In Figure 7.26, the Source Host forwards the packet destined for the Destination Host to Router 1.

Router routing occurs when a router receives a packet that is to be forwarded. The packet is forwarded between routers (when the destination network is not directly attached to the router) or between a router and the destination host (when the destination network is directly attached). In Figure 7.26, Router 1 forwards the packet to Router 2. Router 2 forwards the packet to the Destination Host.

**Router**

Any computer with more than one network interface and which has hardware or software that performs routing is called a **router**. The primary reason for putting a router in a network is to reduce the amount of broadcast traffic by splitting the network into at least two parts.

![Routing Process Diagram](image)

**Fig. 7.26 Routing Process**

### 7.17 Categorising the Computer Networks

One way to categorize the different types of computer network designs is by their scope or scale. For historical reasons, the networking industry refers to nearly every type of design as some kind of area network. Common examples of area network types are:

- LAN - Local Area Network
- WLAN - Wireless Local Area Network
- WAN - Wide Area Network
- MAN - Metropolitan Area Network
- SAN - Storage Area Network, System Area Network, Server Area Network, or sometimes Small Area Network
- CAN - Campus Area Network, Controller Area Network, or sometimes Cluster Area Network
- PAN - Personal Area Network
- DAN - Desk Area Network

LAN and WAN were the original categories of area networks, while the others have gradually emerged over many years of technology evolution.

Note that these network types are a separate concept from network topologies such as bus, ring and star.

**LAN - Local Area Network**

A LAN connects network devices over a relatively short distance. A networked office building, school, or home usually contains a single LAN, though sometimes one building will contain a few small LANs (perhaps one per room), and occasionally a LAN will span a group of nearby buildings. In TCP/IP networking, a LAN is often but not always implemented as a single IP subnet.

In addition to operating in a limited space, LANs are also typically owned, controlled, and managed by a single person or organization. They also tend to use certain connectivity technologies, primarily Ethernet and Token Ring.

**WAN - Wide Area Network**

As the term implies, a WAN spans a large physical distance. The Internet is the largest WAN, spanning the Earth.

A WAN is a geographically-dispersed collection of LANs. A network device called a router connects LANs to a WAN. In IP networking, the router maintains both a LAN address and a WAN address.

A WAN differs from a LAN in several important ways. Most WANs (like the Internet) are not owned by any one organization but rather exist under collective or distributed ownership and management. WANs tend to use technology like ATM, Frame Relay and X.25 for connectivity over the longer distances.
LAN, WAN and Home Networking

Residences typically employ one LAN and connect to the Internet WAN via an Internet Service Provider (ISP) using a broadband modem. The ISP provides a WAN IP address to the modem, and all of the computers on the home network use LAN (so-called private) IP addresses. All computers on the home LAN can communicate directly with each other but must go through a central gateway, typically a broadband router, to reach the ISP.

Other Types of Area Networks

While LAN and WAN are by far the most popular network types mentioned, you may also commonly see references to these others:

- **Wireless Local Area Network** - a LAN based on WiFi wireless network technology
- **Metropolitan Area Network** - a network spanning a physical area larger than a LAN but smaller than a WAN, such as a city. AMAN is typically owned and operated by a single entity such as a government body or large corporation.
- **Campus Area Network** - a network spanning multiple LANs but smaller than a MAN, such as on a university or local business campus.
- **Storage Area Network** - connects servers to data storage devices through a technology like Fibre Channel.
- **System Area Network** - links high-performance computers with high-speed connections in a cluster configuration. Also known as Cluster Area Network.

### 7.18 Working of LAN

A Local Area Network is a small network which is usually contained within one building or campus. It is usually a private network, unlike the public internet. An Administrator in charge controls file sharing, access and many other factors. LANs can be connected to public networks like the Internet, with some precautions (against hackers, viruses etc). Usually a firewall/proxy server/router acts as the gateway between the LAN and the Public Network. A popular wired LAN technology is the Ethernet (Sometimes called IEEE 802.3). These days Wireless LANs are becoming popular. They are collectively known as IEEE 802.11 LANs.

Basically the proxy server will go to internet to pull a webpage for the client requesting it. It also will store a copy of this page (cache) for future requests.
Another function of the proxy is that it hides the client's IP address from the "outside world", and uses its own. Therefore, the webmaster of the webpage can't see the IP of the client requesting the page because in actuality the proxy is requesting the page.

**LAN**

Lan stands for the local area network, i.e., if you are connecting your computer or devices (printer) in a specific limited area, i.e., locally.

A LAN allows certain computers on the network to offer their resources (hard disks, floppy disks, CD-ROMs, Printers, Modems, etc.) for use by other computers on the network as if they were their own. Computers that offer resources are called Servers.

Computers called Workstations can attach the resources (typically hard disks and printers) offered by servers as if they were their own. For instance, at AA Company, computer #1 has a C: hard disk and a D: CD-ROM. Computer #2 has a C: hard disk and a D: CD-ROM, but computer #2 also attaches computer #1's C: drive as its own F: drive. To the user of computer #2, it looks as if drive F: is in his own computer. He can use files and programs from the F: drive just as he can from his own C: drive. The network software module that performs this slight of hand is called the redirector.

A computer can be both a Server and Workstation at the same time, in which case it is called a Peer. Networks without dedicated servers are called peer-to-peer networks. Networks with one or more dedicated servers are called server-based networks even though they may also have peers on them.

Back to our example. The network computers #1 and #2 are on has a server, computer #3. Computers #1 and #2 each have a copy of an accounting program on them, but both read and write accounting data to their G: drive, which is actually C: on computer #3. The tape backup unit is on computer #3 and backs up all the accounting data for all the computers every night by backing up its own C: drive.

When computers #1 and #2 are using the accounting software that software is running in their own memories. The server is not involved at all except to offer its hard disk for data storage. This server is called a file server.

Since this is a Windows accounting package it is big and slow and swaps to disk a lot, so each computer has it installed on its own hard disk to get decent performance. In the days of small fast DOS programs, workstations would also load the program from the server, so it only had to be installed once in one
place.

When computers #1 and #2 do sorted reports, every record has to be read from the server and sorted in the memory of the workstation and written back to temporary files on the server. This causes a lot of network traffic on a larger network.

Lets say AA Company grows a lot and now still has computers #1, #2 and server #3 but has added additional workstations #4 through #29 - and lots of users of the accounting software. all that network traffic causes the network to get really bogged down and users start to complain.

What AA Company does now is ditch that Windows accounting package and install a new multiprocessor Compaq server running Windows NT. The new accounting package uses the Oracle database program to store its data at the server. This new package actually runs on the server (which is now called an application server because it has applications programs running on it). The workstations just have a client program that asks for records and has input and viewing screens. If a client asks for a sorted report all the work is done at the server, cutting network traffic way down. This is called a Client Server network.

Meanwhile, across town, BX Company started with its accounting on a Xenix host computer with some “green screen” terminals wired to it (instead of PCs like AA Company used). There was no network at all, just a lot of serial cables connecting dumb terminals and printers to the host computer.

As it grew, BX upgraded to a Unix host computer and added some PCs that were also wired back to the host and ran terminal emulation software so they could act as terminals to use the accounting. Some of the PCs also got their own printers, which also act as slave printers to the Unix box. Still no network.

Finally, BX Company needed to exchange marketing and project files among the PCs, so they installed a peer-to-peer network connecting all the PCs, and included their big honk’n Sun Enterprise Unix box in the network too. The PCs dumped the terminal emulation package and use telnet which allows them to act as terminals over the network - no more serial cables. Later they add a Linux box to the network to act as a file server, as an Intranet Web server and as a firewall for their DSL connection to the Internet.

### 7.19 Understanding of Blue Tooth Technology

Bluetooth is a high-speed, low-power microwave wireless link technology, designed to connect phones, laptops, PDAs and other portable equipment together with little or no work by the user.
Bluetooth is the name for a short-range radio frequency (RF) technology that operates at 2.4 GHz and is capable of transmitting voice and data. The effective range of Bluetooth devices is 32 feet (10 meters). Bluetooth transfers data at the rate of 1 Mbps, which is from three to eight times the average speed of parallel and serial ports, respectively. It is also known as the IEEE 802.15 standards. It was invented to get rid of wires. Bluetooth is more suited for connecting two point-to-point devices, whereas Wi-Fi is an IEEE standard intended for networking.

**Why is the technology called Bluetooth**

The heart of the Bluetooth brand identity is the name, which refers to the Danish king Harald “Bluetooth” Blaatand who unified Denmark and Norway. In the beginning of the Bluetooth wireless technology era, Bluetooth was aimed at unifying the telecom and computing industries.

Bluetooth can be used to wirelessly synchronize and transfer data among devices. Bluetooth can be thought of as a cable replacement technology. Typical uses include automatically synchronizing contact and calendar information among desktop, notebook and palmtop computers without connecting cables. Bluetooth can also be used to access a network or the Internet with a notebook computer by connecting wirelessly to a cellular phone.

**Types of Bluetooth Devices**

**Bluetooth Dongle**

Bluetooth Dongle: Installing a Bluetooth dongle is easy; simply insert the CD that came with it, follow the on screen prompts and then plug the dongle into a free USB port. If you had a Bluetooth compatible laptop you could just plug the dongle into an internet enabled personal computer and check your e-mail, download Windows updates, or transfer files. On the same lines you could also synchronize your PDA with your personal computer and download the latest appointments, e-mails or send text messages.
**Bluetooth Headset**

Bluetooth Headset: Bluetooth headsets are mainly used with compatible cell phones, place the headset on your ear and roam freely while talking to colleagues, friends and family. You could also connect to a dongle on a personal computer and use it for voice conferencing for example. A number of products exist on the market today, which all offer good sound quality and have a similar variety of features. Prices vary depending on manufacturer but usually you can get a decent one for around Rs. 3700 - 6800.

### 7.20 Role of DNS Server

Domain Name System (DNS) is a system for naming computers and network services that is organized into a hierarchy of domains. TCP/IP networks, such as the Internet, use DNS to locate computers and services through user-friendly names.

To make using network resources easier, name systems such as DNS provide a way to map the user-friendly name for a computer or service to other information that is associated with that name, such as an IP address. A user-friendly name is easier to learn and remember than the numeric addresses that computers use to communicate over a network. Most people prefer to use a user-friendly name—for example, sales.fabrikam.com—to locate an e-mail server or Web server on a network rather than an IP address, such as 157.60.0.1. When a user enters a user-friendly DNS name in an application, DNS services resolve the name to its numeric address.

A DNS server provides name resolution for TCP/IP-based networks. That is, it makes it possible for users of client computers to use names rather than numeric IP addresses to identify remote hosts. A client computer sends the name of a remote host to a DNS server, which responds with the corresponding IP address. The client computer can then send messages directly to the remote host’s IP address. If the DNS server does not have an entry in its database for the remote host, it can respond to the client with the address of a DNS server that is more likely to have information about that remote host, or it can query the other DNS server itself. This process can take place recursively until either the client computer receives the IP address or it is established that the queried name does not belong to a host within the specific DNS namespace.

The DNS server in the Windows Server® 2008 operating system complies with the set of Requests for Comments (RFCs) that define and standardize the DNS protocol. Because the DNS Server service is RFC-compliant and it can use standard DNS data file and resource record formats, it can work successfully...
with most other DNS server implementations, such as DNS implementations that use the Berkeley Internet Name Domain (BIND) software.

In addition, the DNS server in Windows Server 2008 provides the following special benefits in a Windows®-based network:

· **Support for Active Directory® Domain Services (AD DS)**

  DNS is required for support of AD DS. If you install the Active Directory Domain Services role on a server, you can automatically install and configure a DNS server if a DNS server that meets AD DS requirements cannot be located.

  DNS zones can be stored in the domain or application directory partitions of AD DS. A partition is a data container in AD DS that distinguishes data for different replication purposes. You can specify in which Active Directory partition to store the zone and, consequently, the set of domain controllers among which that zone’s data will be replicated.

  In general, use of the Windows Server 2008 DNS Server service is strongly recommended for the best possible integration and support of AD DS and enhanced DNS server features. You can, however, use another type of DNS server to support AD DS deployment.

· **Stub zones**

  DNS running on Windows Server 2008 supports a zone type called a stub zone. A stub zone is a copy of a zone that contains only the resource records that are necessary to identify the authoritative DNS servers for that zone. A stub zone keeps a DNS server hosting a parent zone aware of the authoritative DNS servers for its child zone. This helps maintain DNS name-resolution efficiency.

· **Integration with other Microsoft networking services**

  The DNS Server service provides integration with other services, and it contains features that go beyond the features that are specified in the DNS RFCs. These features include integration with other services, such as AD DS, Windows Internet Name Service (WINS), and Dynamic Host Configuration Protocol (DHCP).

· **Improved ease of administration**

  The DNS snap-in in Microsoft Management Console (MMC) offers a graphical user interface (GUI) for managing the DNS Server service. Also, there are several configuration wizards for performing common server administration tasks. In addition to the DNS console, other tools are provided to help you better manage and support DNS servers and clients on your network.
- **RFC-compliant dynamic update protocol support**

Clients can use the DNS Server service to dynamically update resource records, based on the dynamic update protocol (RFC 2136). This improves DNS administration by reducing the time needed to manage these records manually. Computers running the DNS Client service can register their DNS names and IP addresses dynamically. In addition, the DNS Server service and DNS clients can be configured to perform secure dynamic updates, a capability that enables only authenticated users with appropriate rights to update resource records on the server. Secure dynamic updates are available only for zones that are integrated with AD DS.

- **Support for incremental zone transfer between servers**

Zone transfers replicate information about a portion of the DNS namespace among DNS servers. Incremental zone transfers replicate only the changed portions of a zone, which conserves network bandwidth.

- **Conditional forwarders**

The DNS Server service extends a standard forwarder configuration with conditional forwarders. A conditional forwarder is a DNS server on a network that forwards DNS queries according to the DNS domain name in the query. For example, a DNS server can be configured to forward all the queries that it receives for names ending with sales.fabrikam.com to the IP address of a specific DNS server or to the IP addresses of multiple DNS servers.

### 7.21 Transferring of Email

**Simple Mail Transfer Protocol (SMTP)** is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks.

While electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically only use SMTP for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) or a proprietary system (such as Microsoft Exchange or Lotus Notes/Domino) to access their mail box accounts on a mail server.

Email is submitted by a mail client (MUA, mail user agent) to a mail server (MSA, mail submission agent) using SMTP on TCP port 587. Most mailbox providers still allow submission on traditional port 25. From there, the MSA delivers the mail to its mail transfer agent (MTA, mail transfer agent). Often, these two agents are just different instances of the same software launched with
different options on the same machine. Local processing can be done either on
a single machine, or split among various appliances; in the former case, involved
processes can share files; in the latter case, SMTP is used to transfer the message
internally, with each host configured to use the next appliance as a smart host.
Each process is an MTA in its own right; that is, an SMTP server.

The boundary MTA has to locate the target host. It uses the Domain name
system (DNS) to look up the mail exchanger record (MX record) for the
recipient’s domain (the part of the address on the right of @). The returned MX
record contains the name of the target host. The MTA next connects to the
exchange server as an SMTP client. (The article on MX record discusses many
factors in determining which server the sending MTA connects to.)

Once the MX target accepts the incoming message, it hands it to a mail
delivery agent (MDA) for local mail delivery. An MDA is able to save messages
in the relevant mailbox format. Again, mail reception can be done using many
computers or just one — the picture displays two nearby boxes in either case.
An MDA may deliver messages directly to storage, or forward them over a
network using SMTP, or any other means, including the Local Mail Transfer
Protocol (LMTP), a derivative of SMTP designed for this purpose.

Once delivered to the local mail server, the mail is stored for batch retrieval
by authenticated mail clients (MUAs). Mail is retrieved by end-user applications,
called email clients, using Internet Message Access Protocol (IMAP), a protocol
that both facilitates access to mail and manages stored mail, or the Post Office
Protocol (POP) which typically uses the traditional mbox mail file format or a
proprietary system such as Microsoft Exchange/Outlook or Lotus Notes/Domino.
Webmail clients may use either method, but the retrieval protocol is often not a
formal standard.

SMTP defines message transport, not the message content. Thus, it defines
the mail envelope and its parameters, such as the envelope sender, but not the
header or the body of the message itself. STD 10 and RFC 5321 define SMTP
(the envelope), while STD 11 and RFC 5322 define the message (header and
body), formally referred to as the Internet Message Format.

**File Transfer Operation (FTP) in Web application**

The SocketTools File Transfer product includes both a managed .NET
component and ActiveX control that provides a simplified interface for uploading
and downloading files. By providing a common interface for dealing with FTP
and HTTP servers, the component allows a developer to concentrate on file
transfer, and ignore protocol-specific issues. The component can transfer a file
with a single method, simply by specifying a URL, as in a web browser, without
the need to separately supply the protocol, host, port, file name, and account information. Alternatively, connection and access information may be supplied separately, to allow multiple file transfer operations to be performed in a single server session. In either case, the differences between the supported protocols are kept to a minimum. Advanced features such as support for proxy connections and secure, encrypted client sessions are easily implemented by simply setting a few properties.

**Comprehensive Interface**

The File Transfer component offers a comprehensive interface for FTP, FTPS, SFTP, HTTP and HTTPS, providing the developer with everything that he needs to incorporate file transfers in an application, as well as remote file management. In addition to downloading and uploading by file name, URL, and wild card patterns, a developer may use the component to create directories, search for files and perform other routine file management tasks. The set of properties and methods is sufficiently rich to enable a developer to take advantage of features such as the resumption of interrupted transfers, passive mode operation in the presence of firewalls, automatic file verification and support for custom server commands.

**Security and Reliability**

The SocketTools File Transfer component provides industry standard security using the Secure Sockets Layer (SSL), Transport Layer Security (TLS) and Secure Shell (SSH) protocols. Our components support strong, commercial grade encryption up to 256 bits, and we don’t rely on WinInet or third-party toolkits which have licensing restrictions or are encumbered by patents. Set a single property or option, and the control automatically handles all of the complex certificate management, protocol negotiation and encryption for you. Even advanced options, such as using client certificates, are handled easily with just a few lines of code.

**Internet Standards**

The File Transfer component implements the standard Internet protocols for FTP and HTTP, ensuring the highest degree of compatibility possible with servers of those types around the world. These standards are defined by the Internet Engineering Task Force and published as standard track Request For Comment (RFC) documents. RFC 959 documents the File Transfer Protocol (FTP), which is used for file transfer between a client and a server, and for remote management of files on a server. The Internet draft document “Securing FTP with TLS” describes a mechanism that can be used by FTP clients and servers to implement security and authentication using the TLS protocol defined
by RFC 2246 and the extensions to the FTP protocol defined by RFC 2228. RFC 1945 documents Version 1.0 of the HyperText Transfer Protocol (HTTP), and RFC 2616 documents Version 1.1 of the protocol. These standards govern the communication of client applications such as browsers with web servers. The File Transfer Control implements the GET, POST and PUT commands of these standards.

7.23 Working of Web Server

Whenever you view a web page on the internet, you are requesting that page from a web server. When you type a URL into your browser (for example, “http://www.quackit.com/html/tutorial/index.cfm”), your browser requests the page from the web server and the web server sends the page back:

![Fig. 7.28 Process in Web Server](image)

The above diagram is a simplistic version of what occurs. Here’s a more detailed version:

1. Your web browser first needs to know which IP address the website “www.quackit.com” resolves to. If it doesn’t already have this information stored in it’s cache, it requests the information from one or more DNS servers (via the internet). The DNS server tells the browser which IP address the website is located at. Note that the IP address was assigned when the website was first created on the web server.

2. Now that the web browser knows which IP address the website is located at, it can request the full URL from the web server.
3. The web server responds by sending back the requested page. If the page doesn’t exist (or another error occurs), it will send back the appropriate error message.

4. Your web browser receives the page and renders it as required.

When referring to web browsers and web servers in this manner, we usually refer to them as a client (web browser) and a server (web server).

**Multiple Websites**

A web server can (and usually does) contain more than one website. In fact, many hosting companies host hundreds, or even thousands of websites on a single web server. Each website is usually assigned a unique IP address which distinguishes it from other websites on the same machine. This IP address is also what the DNS server uses to resolve the domain name.

It is also possible to configure multiple websites without using different IP addresses using host headers and/or different ports. This can be useful in a development environment and is quite easy to do.

**Page Not Found**

If the requested page isn’t found, the web server sends the appropriate error code/message back to the client.

You can create user friendly error messages, then configure your web server to display that page instead of the usual error page. This can add a nice touch to your website. How many times have you (or even worse, your visitors) encountered a plain white page with some cryptic error message on it?

It’s very easy to create custom error pages, then configure your web server to use them.

**Default Documents**

If you’ve ever created a website, you may have found that if you have an “index” file (index.html for example), you don’t need to specify the name of the file. For example, the following URLs both load the same page:


In this example, “index.cfm” is the default document. You can configure your web server so that any file name can be the default document. For example, you could configure your web server to use “index.cfm” in the event no filename
has been specified, or if you use PHP, “index.php”. You could even specify different default documents for different directories if you like.

**SSL Certificates**

You can apply SSL certificates against a website via the web server. First you need to generate the certificate either by yourself (i.e. using a certificate generator), or by a Certificate Authority (CA). Then, once it has been generated, you apply it to your website via your web server. Applying an SSL certificate to a website is a straightforward task.

Once you’ve applied an SSL certificate against a website, you can navigate it using HTTPS (as opposed to HTTP). HTTPS encrypts any data that is transferred over the internet. This reduces the possibility of some malicious person being able to read your users’ sensitive information.

### 7.24 Web Browser

Web browser is software application that enables a user to display and interact with text, images, and other information typically located on a Web page at a website on the World Wide Web or a local area network.

Structure of a web page is actually not the way it is displayed in a Web browser. A web page is written in a coded form in HTML, PHP or any other language. Web browser gets this information and formats into the display which we usually see when we visit a webpage. Because of inherent differences in browsers the displayed page might appear slightly different in different browsers.

If you wish to see how a webpage is actually written, do a right click and choose ‘view source’. You will see a clutter commands and text which do not make any sense at all. The browser converts this clutter into understandable display.

Like most of people, you might have used one or two web browsers. But there are many of them available. Internet Explorer is the most common browser used by most of the people. Other web browsers are Mozilla Firefox, Safari, Opera, and Netscape.

Although browsers are typically used to access the World Wide Web, they can also be used to access information provided by Web servers in private networks. They can also be used to access content in file systems like ebooks etc.

Web browsers communicate with Web servers primarily using HTTP (hypertext transfer protocol) to fetch webpages. HTTP allows Web browsers to submit information to Web servers as well as fetch Web pages from them.
Web pages are located by means of a URL (uniform resource locator) which is treated as an address, beginning with http: for HTTP access. Many browsers also support a variety of other protocols, such as ftp: for FTP (file transfer protocol), rtsp: for RTSP (real-time streaming protocol - a protocol for use in streaming media systems), and https: for HTTPS (an SSL encrypted version of HTTP - used to indicate a secure HTTP connection).

In addition to HTML, PHP and other languages, the Web browser also supports various image formats like JPEG, PNG and GIF. The combination of HTTP content type and URL protocol specification allows Web page designers to embed images, animations, video, sound, and streaming media into a Web page, or to make them accessible through the Web page.

### 7.25 Hyperlinks

In computing, a **hyperlink** (or **link**) is a reference to data that the reader can directly follow, or that is followed automatically. A hyperlink points to a whole document or to a specific element within a document. Hypertext is text with hyperlinks. A software system for viewing and creating hypertext is a hypertext system, and to create a hyperlink is to hyperlink (or simply to link). A user following hyperlinks is said to navigate or browse the hypertext.

A hyperlink has an anchor, which is the location within a document from which the hyperlink can be followed; the document containing a hyperlink is known as its **source** document. For example, in an online reference work such as Wikipedia, many words and terms in the text are hyperlinked to definitions of those terms. Hyperlinks are often used to implement reference mechanisms, such as tables of contents, footnotes, bibliographies, indexes, letters and glossaries.

In some hypertext, hyperlinks can be bidirectional: they can be followed in two directions, so both ends act as anchors and as targets. More complex arrangements exist, such as many-to-many links.

The effect of following a hyperlink may vary with the hypertext system and may sometimes depend on the link itself; for instance, on the World Wide Web, most hyperlinks cause the target document to replace the document being displayed, but some are marked to cause the target document to open in a new window. Another possibility is transclusion, for which the link target is a document fragment that replaces the link anchor within the source document. Not only persons browsing the document follow hyperlinks; they may also be followed automatically by programs. A program that traverses the hypertext, following each hyperlink and gathering all the retrieved documents is known as a Web spider or crawler.
Remote login, however, uses simple desktop sharing software to give you a “remote control” for accessing your computer — and all of its software and hard drive files — from any Internet-connected device anywhere in the world.

Remote login works exactly the same way as desktop sharing. In desktop sharing, there are two separate parties: the host computer and the remote user. To share a desktop, the host computer allows a remote user to view the contents of the host computer’s desktop over the Internet. The host computer can also hand over keyboard and mouse controls to the remote user. With remote log-in, your home or work computer is the host and you (in this case) are the remote user.

Remote login requires three basic components:

1. Software download
2. Internet connection
3. Secure desktop sharing network

For remote login to work, both the host computer and all remote users have to download and install the same desktop sharing software. Desktop sharing software typically includes two distinct programs:

1. The desktop sharing client that runs on the host computer
2. A viewer program that allows the remote user to view the contents of the host computer’s desktop in a resizable window

Remote login will only work if the host computer is powered on, connected to the Internet and running the desktop sharing software. Each time you open and run the desktop sharing software on the host computer, the software starts a new session. Each session has a particular ID and/or password that’s required to remotely log in to the host computer. Once the session has been established, most desktop sharing software quietly runs in the background of the host computer until a remote login request is made.

To log in to the host computer from home (or while traveling), you’ll need to run your version of the same desktop sharing software and enter in the correct session ID or password. Or some services allow you to log in through a Web site. Once you’re logged in, both computers will communicate with each other over a secure desktop sharing network. Access to this network can be free or subscription-based, depending on the service. While connected, you’ll have
access to keyboard controls, mouse controls, all software and all files on the host machine.

For security purposes, all packets of information that are sent over the network are typically encrypted on each end with secure shell (SSH) or 128-bit advanced encryption standard (AES) encoding. For added security, no session IDs or passwords are stored on desktop sharing servers; they’re automatically generated by the host machine.

**Short Answer Type Questions**

1. Write any four Components of mother board.
2. What are the slots on the mother board?
3. Write any two Chipset numbers.
4. What are the applications of microprocessor?
5. What are the different types of RAMs?
6. Define Cache memory.
7. What are the applications of AGP card?
8. What are the output voltages of SMPS?
9. What are the supply voltages of HDD, Printer, Motherboard?
10. What are the uses of serial port?
11. What are the uses of Parallel port?
12. What are the applications of USB?
13. Expand SCSI.
14. What are the functions of network card?
15. Expand BIOS.
16. What are the application softwares?
17. What is Networking.
18. Define Router.
20. What are the applications of LAN?
21. What are the applications of Bluetooth?
22. What are the functions of DNS server?
23. What are the applications of Email?
24. What are the applications of FTP?
25. What is Hyperlink?
26. What is the concept of remote login?

### Long Answer Type Questions

1. Mention the components of mother board and write functions of each of them.
2. What are the Expansion slots on mother board. Explain briefly?
3. Write Chipsets number and their use.
4. Explain working of AGP.
5. Explain the working of SMPS with block diagram.
6. Explain working of Serial, Parallel and USB ports.
7. Explain working of SCSI hard disk control.
8. Explain BIOS and operating system
9. Explain working of LAN.
10. Explain working of Bluetooth technology.
11. Explain working of DNS server.
12. Explain working of Email transfer.
13. Explain working of File transfer operation in Web application.
15. Explain concept of Remote login.

### On Job Training / Practical Questions

1. Study the Layout of Motherboard.
2. Study the expansion slots on mother board.
3. Study the process specifications and applications.
4. Study the LAN network.
5. Study the Bluetooth technology.
Word is a word processing program that is used to compose, edit, save and print out text documents. It replaces the job of typewriters in smooth and fast manner in addition to having more features. Basically, Word makes possible what a typewriter does, but in an efficient and economical way. The main advantage is, the matter can be changed as many times as we like and in any format without retyping the text or wasting the stationary.

Word came out with word 6.0 version when it was released for Windows 3.1. Subsequent versions of Word are word 7.0 (Word’95), Word 8.0 (word’97) and Word 9.0 (word’2000), Word 2003, Word 2007, Word 2010.

Microsoft Word 2007

Microsoft word 2007 is a software program (Package) designed by Microsoft Corporation, USA for Word processing application. It is one of most popular word processors in the word for the WINDOWS platform on IBM and compatible PC. It helps you to type, edit store, check spellings and print documents. Word 2007 is used to prepare letters, mailing lists, memos, producing reports etc.

The features of Word processing application are

1. You can change whatever you want in the text without retyping the entire text again.
2. Formatting and printing selected text in boldface, italics or underlined.
3. Changing the font and the size of letters of the selected text.
4. Moving or copying selected text to another location within the document or to another document.
5. Printing selected text in subscript or superscript style.
6. Adding animations to selected text.
7. Searching for a particular word or phrase and replacing it with something else.
8. Aligning (left, right, centered and justify) selected text.
9. Adjustable line and character spacing.
10. Adjustable page size and margins.
11. Facility to create multiple column text.
12. Spelling and Grammar Checking facility.
13. Facility to define headers and footers.
14. Facility to create footnotes and end notes.
15. Facility to insert images in your document.
16. Tables can be created and included in documents.
17. Multiple documents/files can be merged.
18. Toolbars contain buttons that makes easier to perform some common tasks by clicking buttons like saving, printing or formatting text.
19. Text boxes are used to keep text and graphics together.
20. Mail-merge facility allows you to print the same letter with different names and addresses that are saved in another file.

**Word Components**

Word 2007 window contain a menu bar and several toolbars. Most of the buttons on these toolbars act as shortcuts to the commands in the menus. The standard window has the following components.
Title Bar

This is the bar at the top of the window containing minimize, restore or maximize and close buttons. This shows the name of the document and type of the program.

Menu Bar

This is usually below the title bar that provides access to the word menu.

Toolbar

Toolbars give quick access to some common commands. Word has several different toolbars. By default, Word displays the standard and formatting toolbars.

Ruler

The ruler shows tab, margin and indent settings. There are two rulers, horizontal and vertical ruler. These are located above and then to the left of the working area.

Working area or Document area

This is space on word where data or text can be entered or typed.

Insertion Point or Cursor

Insertion point is the blinking vertical line on the working area. This is the point where you can start typing text.

Scrollbars

The horizontal and vertical scrollbars let you bring different parts of a document into view. Scrolling is a technique to see top, bottom, left and right unseen text of the document.

Status bar

Status bar located at the bottom of the window. The status bar tells you about the current status of the insertion point (cursor) or other items.

View Buttons

View buttons provide quick access to see documents in different views. These buttons from left are: Normal view, web layout view, print layout view and outline view.

Starting Word 2007

1. Click the Start button at left end of the task bar.
2. Point to Programs in the start menu.
3. In the programs menu, click Microsoft Word.

Creating a Document

When you start Word, a new document window opens. If word already opened follow the steps below menu bar.
1. Click on File menu on the menu bar.
2. Click on New.... option, select Blank document and click on OK button or.
3. Click on the Blank document button on the standard toolbars.

This is blank screen and is similar to a blank page in your note book. This screen is the first page of your document. This is where you start typing. After the first page gets over, Word automatically goes to the next page. Your document can run into as many pages as required.

Basics of Entering Text

The characters you type are inserted in front of a blinking vertical bar called insertion point. To move the insertion point, you want to insert the text. While typing when a word doesn’t fit it automatically flows to the beginning of next line. This features is called word-wrap. To start a new paragraph position the insertion point as the last line of the first paragraph and then press Enter key.

Saving a Document

The document you type exists only in computer memory, a temporary storage area i.e. in the RAM. It is essential that our document should be stored in a permanent storage device such as hard disk or floppy disk for future retrieval. This is referred to as “Saving a Document”.

The text typed using word is kept in files called documents. Word provides two commands in the file menu to save a document namely. Save and Save as.

Save as: We can use this option to

Save a document for the first time.
Save a previously saved document with a new name.
Save it in a different location.

Save: This option allows saving documents with its existing name. The location and file format not changed.
Procedure for First Time Saving

1. Click **File** menu on the menu bar.
2. Select **Save** or **Save as ...** or click **Save button** on standard toolbar or press **Ctrl + S**, then Save As dialog box appears.
3. Select the folder under which it is saved if required.
4. Type **File name** in the filename text box.
5. Click **Save** button.

Subsequent Saving

1. Click **File** menu on the menu bar.
2. Select **Save** or click on **Save button** on the standard toolbar or press **Ctrl + S**.

Opening an Existing Document

To open existing document

1. Choose **Open...** from the File menu. The open dialog box will be displayed.
2. Select the desired Drive and Directory in the list box.
3. Choose the desired file.
4. Click on **OK** or simply double click the desired file.

Editing a Text

The process of changing the appearance of the text, deleting a portion of the text, duplicating or moving a part of the text, is called Editing. To make these changes you must first select the portion of text, which needs the change.

Selecting Text

In Word, it is very important to know how to select (highlight) text because you can do so many operations on the selected text.

Mouse selection Shortcuts

<table>
<thead>
<tr>
<th>To select this</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>A single word</td>
<td>Double click the word.</td>
</tr>
<tr>
<td>A sentence</td>
<td>Press Ctrl key and click anywhere in the</td>
</tr>
</tbody>
</table>
sentence.

A line of text  
Click in the selection bar area before the line.

Multiple lines  
Click and drag in the selection bar.

A paragraph  
Double click in the selection bar to text to the left of the paragraph.

Any amount of text  
Click at the beginning of the text you want to select, and then shift click at the end of the text.

Whole document  
Triple-click or Ctrl+click any where in the selection bar Edit a Select All command.

Moving Text

1. Select the text to be moved.

2. Select It a **Cut** option or click on **Cut** button on toolbar or press Ctrl+X.

3. Click in a new place, where the selected text should appear.

4. Select **Edit a Paste** or click on **Paste** button on the toolbar or press Ctrl+V.

Copying Text

5. Select the text to be copied.

6. Select **Edit Copy** option or click on **copy** button on toolbar or press Ctrl + C.

7. Click in a new place, where the selected text should copied.

8. Select **Edit a Paste** or click on **Paste** button on the toolbar or press Ctrl +V.

Undo Redo and Repeat

Word 2007 remembers all your actions. At any stage, you can undo not only the latest action also the number of earlier actions related to editing and formatting document. **Undo** cancel (undoes) the last action that you have just performed. **Redo** redoes the last action that you have just undone it. **Repeat** repeats the latest action that you have just done it.

1. To undo the previous work select, **Edit a Undo** or press Ctrl +Z.
2. To redo the previous action select, **Edit a Redo** or press **Ctrl+Y** or press **F4**.

**VIEWS**

Documents in word window can be seen in different ways to make things easier to visualize and faster to work with. There are several ways to visualize documents. These are called views. Microsoft word has seven ways.

1. Normal View
2. Web Layout View
3. Print Layout View
4. Outline View
5. Full Screen View
6. Zoom
7. Print Preview View

There is a **View** option on the menu bar that has all above options except print preview. Print preview is on **File** menu. There are four view buttons in the lower left corner of Word’s status area that are useful to switch views quickly.

**1. Normal View**

This view responds faster to typing, editing, and scrolling. In this view page headers, footers and margins are hidden so that you can concentrate on text. Automatic page breaks are shown as horizontal dotted lines. Normal view displays only the horizontal ruler not the vertical ruler.

**To see document in Normal View**

(a) Select **Normal view** option from the **View** menu.
(b) Clicking **Normal View button** above left-corner of the status bar.
(c) Press **Alt + V** and **N** from keyboard.

**2. Web Layout View**

This view displays documents, as it will look on the web. When a document is open in Web Layout View, Word enlarges the size of text and wraps lines of text so everything is on the screen without having to scroll horizontally.

**To see document in Web Layout View**

(a) Select **Web Layout View** option from the **View** menu.
b) Clicking **Web Layout View button** above left corner of the status bar.

c) Press **Alt + V** and **W** from keyboard.

3. **Print Layout View**

This view shows how text, graphics, and other elements are positioned on the printed page. This is helpful in editing headers and footers, adjusting margins, and with columns and drawings objects.

**To see document in Print Layout View**

a) Select **Print Layout View** option from the **View** menu.

b) Clicking **Print Layout View button** above left corner of the status bar.

c) Press **Alt + V** and **P** from keyboard.

4. **Outline View**

This view shows the document in hierarchical form, headings, subheadings and text under the headings. This view is more helpful than any other in rearranging the documents in specified levels.

**To see document in Outline View**

a) Select **Outline View** option from the **View** menu.

b) Clicking **Outline View button** above left corner of the status bar.

c) Press **Alt + V** and **O** from keyboard.

5. **Full Screen View**

This view removes all the menus and toolbars as well as takes up all available space in the Word window. To see a document in Full Screen, click **View** menu and select **Full Screen** option or press **Esc**. Key on the keyboard to return to the previous view. Keyboard shortcut **Alt + V + U**.

6. **Zoom**

This view enlarges or reduces the view of your document as a percentage of its normal size. To return your document to normal size, click **1000%**. To zoom a document click view on the menu bar select Zoom, in the zoom dialog box select zoom setting that you want and then click **OK** button. Keyboard shortcut **Alt + V + Z**.
7. Print Preview View

This view shows exactly how it will look when it is printed on. When this view is selected Word resizes the document so it fits inside the document window. To see a document in this mode select **Print Preview** from file menu bar or click **Print Preview button** on **Standard Toolbar.** Keyboard shortcut is **Alt + F + V.**

**Carry Out Spell Check**

Word offers tools that easier to examine and automatic your writing tasks. Spelling and Grammer is a tools used to check the active document for possible spelling, grammer, and writing style errors, and displays suggestions for correcting them.

**Automatically Check Spelling and Grammer as you Type**

As soon as you type misspelled or ungrammatical sentences, word recognizes them to be misspelled and ungrammatical and then underlines them immediately in red and green wavy lines. The red underlines indicate possible spelling errors and the green underlines indicate possible grammatical errors. To fix one of these marked errors, right click on it, you get a shortcut menu that has a set or replacement words, along with a few other options. If you think any one of them is suitable click on it to replace otherwise click ignore all to tell it is not suitable word and need not be replaced.

**Check Spelling and Grammer All at Once**

This method is useful if you want to postpone spell check of a document till the edit is finish. If you want to check a portion of the document, select it. To open spelling and grammer check dialog box.

1. Click Tools menu and select Spelling and Grammer option.
2. Click the Spelling and Grammer button on the standard toolbar.
3. Press F7 on keyboard.

**Spelling and Grammer dialog box Options**

**Symbols and Special Characters**

Symbols and special characters that are not available on the keyboard can be inserted using Symbol feature or keyboard shortcuts. To display the symbol dialog box select Symbol from insert menu or press **Alt + I** and then **S.**
**Insert a Symbol**

1. Click where you want to insert a symbol.
2. On the **Insert** menu click **Symbol**.
3. In the Symbol dialog box select the symbol and click **Insert** button or double click on the symbol.

**Applying Font Color**

1. Select the Text.
2. Click down arrow of the font color button on the formatting toolbar and then select the color from available colors.
3. Click on the **Format** menu and select the option **Font**. In the font dialog box choose the required **color** from the **Font color** list box and then click on **OK** button.

**Applying Animations**

Text animation is a technique used to add visual movements such as blinking, marching, moving, shimmering, sparkling etc.

1. Select the text you want to animate.
2. On the **Format** menu click **Font**, and then click the **Text effects** tab.
3. In the **Animations** box, select the effects you want and click on **OK** button.

**Special Effects**

You can decorate your text, as you want using font dialog box. There are different formatting options in the form of check boxes named under effects. There are

1. **Strikethrough**: Draws a line through the selected text.
2. **Double Strikethrough**: Draws a double line through the selected text.
3. **Superscript**: Raises the selected text above the base line and changes it to a smaller size.
4. **Subscript**: Lowers the selected text below the base line and changes it to a smaller size.
5. **Shadow**: Adds a shadow behind the selected text, beneath and to the right of the text.
6. **Outline:** Displays the inner and outer border of each character.

7. **Emboss:** Makes selected text appear as if it is raised off the page in relief.

8. **Engrave:** Makes selected text appear to be imprinted or pressed into the page.

9. **Small Caps:** Formats selected text as capital letters and reduces their size.

10. **All Caps:** Formats selected text as capital letters.

11. **Hidden:** Prevents selected text from being displayed or printed.

**Change Case**

Change Case changes the capitalization of selected text. It is used to change text from uppercase to lowercase and vice versa.

**Procedure**

1. Selected text to be changed.

2. Choose **Change Case...** option from format menu to displays the change case dialog box. There are five options.
   
   **a) Sentence Case:** Capitalized the first letter of the first word in the selected sentences.

   **b) Lowercase:** Changes all selected text to lowercase letters.

   **c) Uppercase:** Changes all selected to capital letters.

   **d) Title Case:** Capitalizes the first letter of each word in the selected text.

   **e) Toggle Case:** Changes all uppercase letters to lowercase in the selection and vice versa.

3. Select the option you want and click **OK**.

**Paragraph Formatting**

In general paragraph is collection sentences. In word 2000, a paragraph in nothing more than a collection of words that ends when you press Enter Key. When you press the Enter Key, you are marking the end of a paragraph. Paragraph formatting affects individual paragraphs.

**Alignment**

Alignment means an arrangement of text into suitable order in the paragraph.
If shapes whether your text is lined up on the left, right, in the center, or on both left and right so that it leads to smartening the paragraph. There are four buttons on the formatting toolbar namely Align Left, Align Right, Center and Justify.

1. **Align Left**: Aligns the selected text to the left with uneven right edges.
2. **Center**: Centers the selected text.
3. **Align Right**: Aligns the selected text to the right with uneven left edges.
4. **Justify**: Aligns the selected text to both the left and right margins or indents.

**To Change Alignment using Paragraph**

1. Place the insertion point anywhere in the paragraph.
2. Click the alignment (Left, Right, Center, Justify) button on the standard toolbar.

**Keyboard Shortcuts**

1. **Align Left**: Ctrl + L
2. **Align Right**: Ctrl + R
3. **Center**: Ctrl + E
4. **Justify**: Ctrl + J

**Indentation**

Indentation is a process used to leave certain number of blank spaces before the first line, as well as form the right and left margins of paragraph. Indentation can be classified into four categories Left, Right, First Line and Hanging.

1. **Left Indentation**
   
   Left indentation leaves certain number of blank spaces after the left margin.

2. **Right Indentation**
   
   Right Indentation leaves certain number of blank spaces before the right margin.

3. **First Line Indentation**
   
   First Line Indentation leaves certain number of blank spaces of before the first line of the paragraph.

4. **Hanging Indentation**
Hanging Indentation leaves certain number of blank spaces for all line but the first from the left margin of a paragraph.

**Indentation using Indent Markers**

1. Select the paragraph(s) you want to indent.
2. Point to the indent marker that you want to use.
3. Drag the desired indent marker to the right spot on the ruler, and release the mouse button.

**Keyboard Shortcuts**

1. Ctrl + M : Left Indent
2. Ctrl + Shift + M : Remove Left Indent
3. Ctrl + T : Hanging Indent
4. Ctrl + Shift + T : Reduce Hanging Indent
5. Ctrl + Q : Remove Paragraph Formatting

**Line Spacing**

Line spacing determines an amount of vertical spaces among lines in a paragraph. It allows us to expand or condense the vertical spaces lines of paragraph. Line spacing can be set using either the Line spacing box in the paragraph dialog box or keyboard shortcut or buttons in the formatting toolbar.

1. **Single Line Spacing**

   This is Word’s default spacing. It accommodates the largest font in that line, plus a small amount of extra space.

2. **Line-and-a-half Spacing (1.5 lines spacing)**

   This applies one-and-a-half times that of single line spacing.

3. **Double Line Spacing**

   This follows twice that of single lines spacing.

4. **At least Spacing**

   This allows minimum line spacing that Microsoft Word can adjust to accommodate larger font sizes or graphics.

5. **Exactly Spacing**
This allows increasing fixed line spacing that word does not adjust. This option spaces all lines uniformly.

6. Multiple Spacing

This allows increasing or decreasing line spacing by a percentage that you specify.

Applying Lines Spacing

Place the insertion point in the paragraph or select multiple paragraph choose the option required line spacing option from the line spacing drop down list of the paragraph box.

Click on OK button

Keyboard Shortcuts

1. Single Line Spacing : Ctrl + 1
2. 1.5 Line Spacing : Ctrl + 5
3. Double Line Spacing : Ctrl + @

Bullets and Numbering

Bullet is an item used to highlight sections of text into an unordered list, in order to bring attention to them. It is used makes lists of information stand out from the rest of your document text.

Numbering is to organize your information into a series into series that is easy ot understand. Make sure each item in the list or each point is on a separate line.

To insert a Bullet or Numbering at the start of align, just click the Bullets button or Numbering button on the Formatting toolbar.

If you want to put bullets or numbering for created information do the following.

1. Select the items you want to add bullets or numbering to.

2. Click Bullets button to add bullets, click Numbering button to add numbering.

To create a bulleted or numbered list automatically as you type, type 1 or* (asterisk), press SPACEBAR or TAB, and then type any text you want. When you press ENTER to add the next list item, word automatically inserts the next number or bullet. To finish the list press ENTER twice.
To Change the Default Bullet Symbols or Number Styles

1. Click Format on the menu bar.
2. Select Bullet and Numbering.
3. Click the Bullet and Numbering tab.
4. Select 1 to 7 types of bullets or numberings.

To change the layout and style click the Customize command button to get the dialog box. In the dialog box can select a different symbol from any of the fonts available to you.

TAB

Tab helps you line up columns of information or indent text within a line or paragraph. You can add, remove or move tabs as you desire. Each tab position set on the horizontal ruler is called tab or tab stop. When the tab key on the keyboard is pressed the insertion point jumps from one tab to the next. Microsoft Word has five kinds of tab stops given below

1. **Left** (L) : Text extends to the right from the tab stops.
2. **Center** (^) : Text is centered at the tab stop.
3. **Right** (+) : Text extends to the left from the tab stops.
4. **Decimal** (^) : Text before the decimal point extends to the left, and text after the decimal point extends to the right.
5. **Bar** (I) : Insert a vertical line at the tab stop.

The tab selector button at the left end of the horizontal ruler is used to select different tab stops. Click it to cycle through left tab, center tab, right tab, decimal tab and bar tab. The quickest way to set tabs for the current paragraph or paragraphs is to use the horizontal ruler.

Procedure for Setting Tabs with the Horizontal Ruler

1. Click any where in the paragraph or select multiple paragraphs.
2. Click the tab selector button at the left of the horizontal ruler until the tab type you want is come up.
3. Click in the horizontal ruler where you want the tab stop to be.

Procedure for Setting Tabs with the Tabs Dialog Box
1. Click any where in the paragraph or select multiple paragraphs.

2. Click the **Format** menu and select **Tabs...**

3. Click in Tab stop position box and enter a measurement that tells how far the tab stop positioned from the left margin.

4. Select the alignment option for the tab and click the set button.

5. You can select as many tabs you want, and then click **OK** or press **ENTER** key.

**Borders and Shading**

Border means a line drawn in the box shape around the text or page. Borders can be applied any or all sides of each page in a document. You can also put borders around graphics. You can add page borders in many lines styles and colors, as well as a variety of graphical borders.

Shading means filling the background of the text with a color in order to draw people attention to the text. Shading can be added to paragraphs or with or without borders.

Borders and Shading can be added using the Table and Borders toolbar, or the Borders and Shading dialog box.

**Procedure for Applying Borders**

1. Select the text to be applied.

2. Click the **Format** menu and select **Border and Shading...** option.

3. Click the Border or Page Border tab (for page Border).

4. Select the required line style and line weight and line color from the provided list boxes.

5. Select the borders style (None, Box, Shadow etc).

6. Click on OK button.

7. Select the text to be applied.

8. Click the **Format** menu and select **Border and Shading...** option.

9. Click the **Shading** tab and select the shading color and style.

10. Click **OK** button.
PAGE SETUP

Page setting means setting the size of a page, margins (left, right, top, bottom etc), orientation (portrait or landscape) etc. The page setup dialog box helps you to control page layout to a great extent in Word 2000.

Setting of Margins and Indents

Page margins are the blank spaces around the edges of the pages. All four margins namely top, bottom, left, right are adjustable. We can use word’s default page margins or specify our own margins.

Changing Margins using Pages Setup dialog Box

1. Click on **File** menu and select the option **Page Setup**...
2. Click the **Margin** tab if it is not already in front.
3. Click on the up or down arrows (spinner arrows) to the right of the margins boxes to increase or decrease the margins. Or click on the box and type the new value.
4. Click **OK** button.

Setting the Page Size

Word 2007 will enable you to change the paper size to anything you want. When you create a new document, Word picks up a paper size which is usually the Letter size 8.5 inches wide and 11 inches long A4 size.

1. Click on **File** menu and select **Page Setup**... option.
2. Click **Paper Size** tab to bring it to the front.
3. Click the paper size drop down arrow and select the required paper size.
4. Select the page orientation either **Portrait** or **Landscape**.
5. Click **OK** button.

Inserting Page Breaks

Page break is the point at which one page ends and another page begins. Page breaks are used to begin a new page of text before the previous page is filled. Page breaks can be inserted automatically or manually.

When the current page is full, word 2000 automatically starts a new page. These page breaks are called automatic page breaks. The page breaks inserted
manually are called manual page breaks.

**Inserting Page Breaks Manually**

1. Click the spot where you want to insert the break.
2. Click **Insert** menu and select the option **Break...** to display the break dialog box.
3. If necessary, click the page break radio button.
4. Click **OK** button.

**Removing Page Breaks**

1. Click on the page break line and press the **Delete** key or if the insertion point is just after the break, strike the backspace key.

**Inserting Page Numbers**

Microsoft Word provides facility add page numbers to the top or bottom of the document. Follow these steps to add page numbers.

1. Click **Insert** menu and select the option **Page Number..**
2. Choose the **position** (bottom or top) from the position list box.
3. Choose the **alignment** (left, right, center etc) from the alignment list box.
4. Click on **OK**.

**Including Headers and Footers**

Header is an area used to place repetitive information across the top of each page in a document. FOOTERS is an area used to put repetitive information across bottom of each page in a document. We can use the header and footer throughout a document or change the header and footer for part of the document.

**Applying Header and Footer**

1. Click the **View** menu and select **Header and Footer** option. It displays the Header and Footer toolbar and shows the header and footer area of the document.
2. Select the Header and Footer bottom and enter the text. Use the tool buttons. (Insert page number, no.of pages, Format page numbers, Insert date, Insert time etc).
3. Click on Close button on the toolbar.
Multiple Columns

Word allows arranging the text into several columns like newspapers columns. When one column is full the text flows into the next columns. To prepare a column text.

1. Select the text.
2. On the Standard toolbar, click Columns button or select Columns from the Format menu to display the columns dialog box.
3. Drag to select the number of columns on the columns button of the standard toolbar. Or select the number of columns from the columns dialog box.

Tables

A table is collection of rows and columns. Each row and column inside a table is made up of units called cells. A cell is the intersection of rows and columns. A current cell is where you type a table entry. Tables are often used to organize and arrange information.

Creating a Table

Word provides three ways for create a table.

1. Using the Insert Table tool on the standard toolbar.
2. Using the Insert Table dialog box.
3. Using the Tables and Borders toolbar.

1. Creating a Table with “Insert Table” tool
   a) Place the insertion point where you want to insert the table.
   b) Click the insert table button on the standard toolbar.
   c) Drag the mouse pointer down and to the right over the grids and release the mouse pointer.

2. Creating a Table Using the Insert Table dialog box
   a) Click where you want to insert a table.
   b) Click on Table menu, select Insert option and click on Table ... option.
   c) Type or select the number of rows and columns from the appropriate boxes.
d) Click **OK** button

**Adding Cells, Columns and Rows to a Table**

1. Place the insertion where cells, columns or rows are required.
2. Click the table menu and select insert you will see a list of options select the required option.
3. To quickly add a row at the end of a table, click the last cell of the table last row and then press the **TAB** key.

**Delete cells, rows or columns from a Table**

1. Select the cells, rows or columns to be deleted.
2. On the table menu, point to **Delete**, and then click columns, rows or cells.
3. While deleting cells, click the option to be deleted.

**Clear the Contents of a Table**

1. Select the content to be cleared
2. Press **Delete** key.

**Table and its contents**

1. Click in the table to deleted.
2. On the Table menu, point to **Delete**, and then click **Table**.

**Mail Merge**

Mail merge is an ability to create a letter in Word, and then print multiple copies of that letter, each with a different name and address on it. Mail merge in word may also be used to fill envelopes and labels with data as well. Mail merge is the process of taking document and mixing it with another to create an almost identical bunch of new document.

**Mail Merges Features of Word**

**Main Document**

This is the actual document that you are producing. It can be a form letter, label or envelope. The main document contains normal text that stays the same for each copy of the letter.
Data Source

This is the file that store information to be brought into the main document. This file contains a collection of names and address from which the program gets what you want to included in the document.

Performing a Mail Merge involves three basic steps

1. Creating the main document (such as form-letter).
2. Creating the data source (such as address list).
3. Merging the main document with data source.

Creating a Main Document

1. Click Tools menu and select the option Mail Merge...
2. Click the create button located below “Main Document” in step. a) (Word displays a list box that contains the various types of Mail Merge documents you can create).
3. Click Form Letters in the drop down list.
4. Word displays a message box, asking you weather you want to use the active document as your main document or start a new Document. Click Active Document if active document contain form letters.

Creating a Data Source

1. Click the Get Data button. Word displays four options in a list box.
   a) If you want to create a fresh list of addresses then you should click on Create Data Source.
   b) If you want ot open a list of addresses you have created already then you should click on Open Data Source option.
2. In the create data source dialog box, choose the field names and add new fields if required, delete unnecessary fields and click OK button.
3. Then “Save Data Source” dialog box appears. Save the data source file.
4. The Word displays a message box with the options “Edit Data Source” and “Edit Main Document”.
5. Click “Edit Data Source” button.
6. In the Edit Data Source dialog box add records (addresses).
7. Click **OK** button.

8. The main document appears with the new **mail merge toolbar**.

**Inserting Merge Fields in the Main Document**

1. Insert the merge fields in the main document in required places using the Insert Merge Fields button from the Mail Merge toolbar.

2. To view the data records click on View Data Records button (<<ABC>>) on the mail merge toolbar.

3. To merge the Data Source and Main Document click on “**Merge..**” button on the mail merge toolbar.

4. Click on “**Merge**” button in the merge dialog box.

**Printing a Document**

Printing means getting a hard copy of the information a printer on the paper. When a new document is going to be created you must consider the type of printer, paper size and document design used. When the document is ready to be printed see the print preview. To print active file or selected part, just click printer button on the **standard toolbar**. To print document click **File** menu and select the option **print**. In the print dialog box specify the page range and number of copies and click on **OK** button.

---

**8.2 Microsoft Excel 2007**

**Entering Text and Numbers**

Microsoft Excel is an electronic spreadsheet that runs on a personal computer. You can use it to organize your data into rows and columns. You can also use it to perform mathematical calculations quickly. This tutorial teaches Microsoft Excel basics. Although knowledge of how to navigate in a windows environment is helpful, this tutorial was created for the computer novice. This lesson will introduce you to the Excel window. You use the window to interact with Excel.

- The **Microsoft Excel window**
- The **Microsoft Office Button**
- The **Title Bar**
- The **Ribbon**
- **Worksheets**
• The Formula Bar
• The Status Bar
• Move Around a Worksheet
• Go To Cells Quickly
• Select Cells
• Enter Data
• Edit a Text
• Wrap Text Delete a Cell Entry
• Save a File
• Close a Excel

**Entering Excel Formulas and Formatting Data**

Lesson 1 familiarized you with the Excel 2007 window, taught you how to move around the window, and how to enter data. A major strength of Excel is that you can perform mathematical calculations and format your data. In this lesson, you learn how to perform basic mathematical calculations and how to format text and numerical data.

• Set the enter key direction
• Perform Mathematical calculations
• Autosum
• Perform automatic calculations
• Align cell entries
• Perform advanced mathematical calculations
• Copy, cut, paste and cell addressing
• Insert and delete columns and rows
• Create borders
• Merge and center
• Add background color
• Change the font, font size and font color
Lesson 3: Creating Excel Functions, Filling Cells, and Printing

By using functions, you can quickly and easily make many useful calculations, such as finding an average, the highest number, the lowest number, and a count of the number of items in a list. Microsoft Excel has many functions you can use. You can also use Microsoft Excel to fill cells automatically with a series for example, you can have Excel automatically fill your worksheet with days of the week, months of the years, of the types of series.

A header is text that appears at the top of every page of printed worksheet. A footer is text that appears at the bottom of every page of your printed worksheet. You can use a header or footer to display among other things titles, page numbers, or logos. Once you have completed your Excel worksheet, you may want to print it. This lesson teaches you how to use functions, how to create a series, how to create headers and footers, and how to print.

Creating Charts

In Microsoft Excel, you can represent numbers in a chart. On the insert tab, you can choose from a variety of chart types, including columns, line, pie, bar, area, and scatter. The basic procedure for creating a chart is the same no matter what type of chart you choose. As you change your data, your chart will automatically update. This lesson teaches you how to create a chart in Excel.
• Switch data
• Change the style of a chart
• Change the size and position of a chart
• Move a chart to a chart sheet
• Change the chart type

8.3 PowerPoint Basics

The tutorial provides an overview of how to use many of the features of Microsoft Powerpoint. Step-by-step instructions are provided for creating presentations; adding new slides; using spell check; saving a presentation; previewing a presentation; inserting clip art; moving and resizing objects; using graphic images from the internet; moving, copying, and deleting slides; adding transitions; adding animations; applying a template; and printing a presentation.

PowerPoint

Powerpoint is the presentation graphics component of Microsoft Office that lets you create and save presentations. A presentation is a collection of slides relating to a topic which may be shown while an oral report is given. From the slides you can create color or black & white overheads, on-screen presentation and 35 mm slides. To support your presentation you can provide an overview of your presentation.

Starting PowerPoint

To Start PowerPoint

1. Click on the Start button on the left side of the taskbar, then click programs; then click Microsoft PowerPoint.

Creating a Presentation Using the Blank Presentation Option

To Create a Blank Presentation Using the Blank Presentation Option

1. Select Blank Presentation from the PowerPoint dialog box.

2. Accept the Title Slide layout that has been selected for you by clicking OK. A Title slide is then created for you to enter information.

3. Click on inside the “Click to add title” place holder and type the title of your presentation.

4. Click inside the “Click to add sub-title” place holder and type your sub-title.
Adding New Slides

To Add a New Slide to a presentation

1. Click the New Slide button located at the top left of screen on the common tasks toolbar or click Insert on the Menu bar, then New slide.

2. Accept the Bulleted list layout that has been selected for you by clicking OK.

3. Click inside the “Click to add title” place holder and type the title for this slide.

4. Click inside the “Click to add text” placeholder and type the content for the slide.

Notes

♦ Return to Slide one of using one of the following methods:
♦ Use the Page Up key.
♦ Click the previous Slide button on the vertical slide scroll bar.
♦ Drag the vertical slide scroll up until the desired slide number is displayed.

Using Spell Check

To Spell Check Your Presentation

1. Use one of the following methods.

♦ Click the spelling icon on the Standard toolbar.
♦ Select Spelling from the tools main menu.

Saving a Presentation

To Save a Presentation

1. Go to the Menu Bar and click File.

2. Click Save. The Save As dialog box will then appear. Change the directory, if needed, to find the directory you want to save the file in. In the Filename box type the name of the file, then click OK.

Previewing Your Presentation on the Screen

To preview Your Presentation on the screen

1. Click the Slide show icon (bottom left of screen).
2. Click mouse to advance through the presentation slide by slide.

3. Change back to slide when to make changes. Edit any changes by clicking in the appropriate placeholder.

4. **SAVE** any changes made by clicking the save icon on the standard toolbar.

**Existing Powerpoint**

**To Open an Existing File**

1. Click **File**.

2. Click **Open**, or from PowerPoint dialog box, select **Open an Existing Presentation**.

3. Select the file name (Change directory to Desktop if necessary).

4. Click **OK**.

**Inserting Clip Art**

**To Insert Clip Art**

1. Go to slide where you want to insert clip art.

2. Click the layout button at top right of screen in toolbar.

3. From Row 3, click to select layout 2, Clip Art & Text.

4. Click **Apply**.

5. Double-click in the graphic placeholder to add Clip Art.

6. Examine the categories of clip art. Select one of the choices.

7. Click **OK**.

8. Click **Save**.

**Moving and Resizing Objects**

Text, clip, and object placeholders can be moved, copied, sized, and deleted. To do this, you must first display handles to put the placeholder into an edit mode. Click on the text to display the placeholder, then click on the placeholder border to display the handles.

When handles appear, you can size the placeholder or graphic by dragging...
a corner handle to resize proportionally. When you size a text placeholder, the
text within it will adjust to the new borders.

You can move a placeholder and its contents by displaying the handles, then
placing the pointer on the border (not a handle), clicking and holding the left
mouse button while dragging the placeholder to a desired location.

**Using Graphic Images from the Internet**

If you don’t find the image you want in the MS Clip Art Gallery, one good
option is to take images off the World Wide Web. You can save these files and
insert them into your documents and presentations. Make sure your PowerPoint
presentation is saved before you began.

1. Open an Internet browser, such as Internet Explorer or Netscape and
connect to the Internet.
2. Locate an image you want to download.
3. Using your mouse, right-click on the image. A pop up menu with then
appear.
4. Select the **Save Picture As...** menu item.
5. Save the image in a drive and folder where you can find it.
6. Close the browser.
7. Return to PowerPoint and go to the slide where you want to use the
image.
8. Insert the image by going to **Insert**, then **Picture**, then **From File**.
Find the image you download and insert it into your slide.
9. Adjust the size. (See Moving and Resizing Objects).

**Moving, Copying, and Deleting Slides**

**To Move, Copy, or Delete Slides**

1. Switch to slide sorter view.
2. Click to select the slide you want to move.
3. Drag the slide to the position where you want it to be and let go of the
mouse button.
4. **Save**.
5. Copy a slide by holding down the Ctrl key and dragging the slide to the
Presentation of Slide

1. Select **Print** from the **File** main menu or press **Ctrl + P**.

2. Select the desired file settings and click **OK**.

**Note:**

- In the print dialog box which follows, you may print the active slide, a selected slide range, or all slides in a presentation.
- When you print all slides of presentation, each slide prints on a separate page.
- The print what feature allows you to indicate whether you want your presentation printed as slides, notes pages, handouts with 2, 3 or 6 slides per page, or as an outline.

6. Delete a slide by selecting it and then pressing **Delete**.

**Note:** Since you cannot edit contents in slide sorter view, you will need to return to slide view to make changes and adjust text.

**Adding Transitions**

Transitions control the way slides move on and off the screen. They may be added to slides in all views, but slide sorter view offers the quickest and easiest way. However, other views provide a transitions dialog box which will preview the transition you select.

1. Switch to slide view.

2. Click **Slide Show** in the main menu.

3. Click **Slide Transition**.

4. Select a transition type and note its effect in the window on the bottom right of the dialog box. A description of transition types and their effect appears below.

5. Also in the transitions dialog box, you can select the speed of the transition and whether you want to change the slides manually (by a mouse click) or automatically (after a specified number of seconds).

**Adding Animations**

An animation (build) is the style in which each bulleted item and or image
appears on a slide. Powerpoint allows you to specify the slide to contain builds and the build effect. A slide that does not contain an animation will reveal all bulleted items at once during a slide show. A slide that does contain an animation will reveal no bulleted item until you activate each one when you are ready to discuss that point.

1. If necessary switch to slide view.

2. Select Custom Animation from the slideshow menu.

3. Click the Timing tab.

4. Click on the text you want to animation.

5. Click the Effects tab.

6. In the Animation dialog box which follows, you have several options for the way a bulleted item will display on a slide. Make a choice. For example, you may select Text in the animation order box, then select an effect under Entry Animation and sound.

7. To have previous text as new text appears, click the down arrow and select your color for the dimmed items.

8. Save and close the file.

Note: In slide sorter view, slides containing builds and transitions are marked by respective icons below and to the left of the miniature slide image.

Applying a Template

The template option allows you to create slides with a pre-designed format. Powerpoint previous over 100 professionally designed formats with colorful background and text from which you can choose.

1. Click the Apply Design button on the common tasks toolbar.

2. An apply design dialog box then appears which lists the available templates. As you click on each template name, a sample of it appears in the right side of the window.

3. After selecting the template design you sedire and clicking Apply, the template design is appliead to all the slides in the presentation.

4. Save.
Short Answer Type Questions

1. What is Word count?
2. What is default dictionary?
3. What is Thesaurus?
4. What is undo changes?
5. What are different types tab alignments?
6. What are the options in page layout?
7. What is Orientation?

Long Answer Type Questions

1. Write about formatting the paragraph.
2. Write about find and replace.
3. Write the steps to create, save and preview a presentation.
4. Write about formatting a text in power point.