**University of Mosul**
**College of Science**
**Department of Physics**
**Second Stage**
**Lecture** 3

# Digital Electronics

**Lecture 3 : Numbering system and binary Codes**

**أ.م. يسرى مال الله عبدالله**

**2024 - 2025**

## Hexadecimal Numbers

| ثنائي | ثماني | ست عشري | عشري |
|---|---|---|---|
| 0000 | 0 | 0 | 0 |
| 0001 | 1 | 1 | 1 |
| 0010 | 2 | 2 | 2 |
| 0011 | 3 | 3 | 3 |
| 0100 | 4 | 4 | 4 |
| 0101 | 5 | 5 | 5 |
| 0110 | 6 | 6 | 6 |
| 0111 | 7 | 7 | 7 |
| 1000 | 10 | 8 | 8 |
| 1001 | 11 | 9 | 9 |
| 1010 | 12 | A | 10 |
| 1011 | 13 | B | 11 |
| 1100 | 14 | C | 12 |
| 1101 | 15 | D | 13 |
| 1110 | 16 | E | 14 |
| 1111 | 17 | F | 15 |
| 10000 | 20 | 10 | 16 |

الأرقام من 1 ~ 16 عشريا ممثلة بالأنظمة الثنائي والست عشري والثماني

## 1. Hexadecimal Number

The **hexadecimal number system** is used commonly by designers to represent long strings of bits in the addresses, instructions, and data in digital systems. This system uses 16 digits to represent any quantity. The positional weight of each digit is a power of 16.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| …. | $16^4$ | $16^3$ | $16^2$ | $16^1$ | $16^0$ | . | $16^{-1}$ | $16^{-2}$ | $16^{-3}$ | ….. |
| …. | 65536 | 4096 | 256 | 16 | 1 | . | $\frac{1}{16}$ | $\frac{1}{256}$ | $\frac{1}{4096}$ | …. |

Each hexadecimal digit represents a 4-bit binary number (as listed in Table 1).

| Decimal | Binary | Hexadecimal |
|---|---|---|
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| 3 | 0011 | 3 |
| 4 | 0100 | 4 |
| 5 | 0101 | 5 |
| 6 | 0110 | 6 |
| 7 | 0111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

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

**10**, 11, 12, 13, 14, 15, 16, 17, 18, 19,

1A, 1B, 1C, 1D, 1E, 1F,

**20**, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2A, 2B,

2C, 2D, 2E, 2F,

**30**, 31,

## Counting in Hexadecimal

How do you count in hexadecimal once you get to F? Simply start over with another column and continue as follows:
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F,

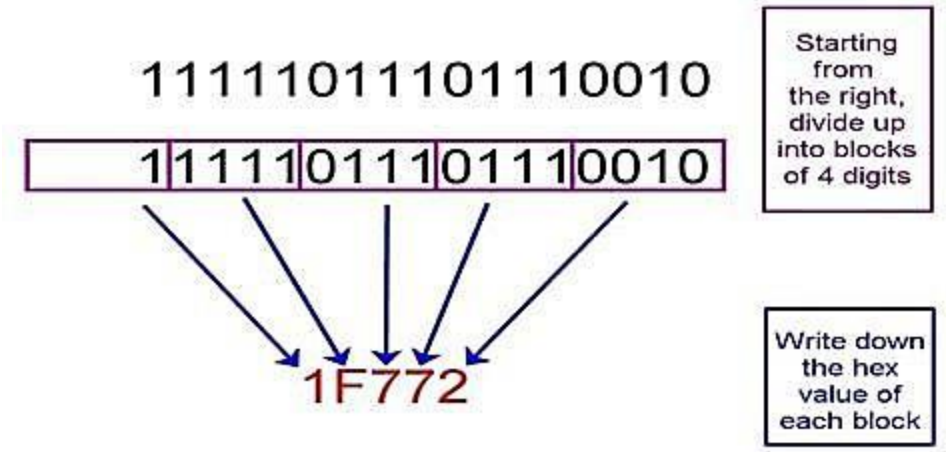
With two hexadecimal digits, you can count up to $FF_{16}$, which is decimal 255. To count beyond this, three hexadecimal digits are needed. For instance, $100_{16}$ is decimal 256, $101_{16}$ is decimal 257, and so forth.

The maximum 3-digit hexadecimal number is $FFF_{16}$, or decimal 4095. The maximum 4-digit hexadecimal number is $FFFF_{16}$, which is decimal 65,535.

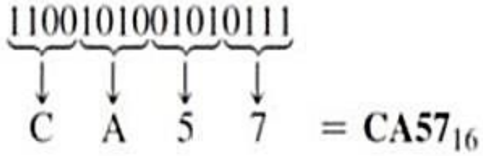
## Number Base Conversion

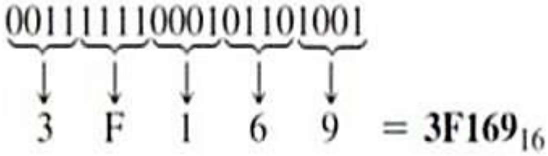
### A. Binary-to-Hexadecimal

**Conversion Examples**




$11111011101110010_2 = 1F772_{16}$



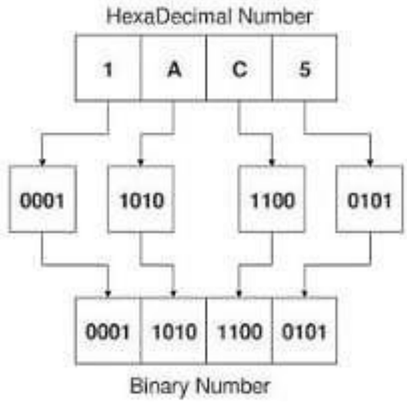
## Hexadecimal-to-Binary Conversion

Hexadecimal-to-Binary Conversion: Replace each hexadecimal symbol with the appropriate four bits.

**Examples**

(a) 1 0 A 4 → $1000010100100$     (b) C F 8 E → $1100111110001110$     (c) 9 7 4 2 → $1001011101000010$

(b)




### B. Hexadecimal-to-Decimal Conversion

1. **Method One:** First convert the **hexadecimal number to binary** and then convert from **binary to decimal.**

(a) 1 C → $00011100 = 2^4 + 2^3 + 2^2 = 16 + 8 + 4 = \mathbf{28_{10}}$

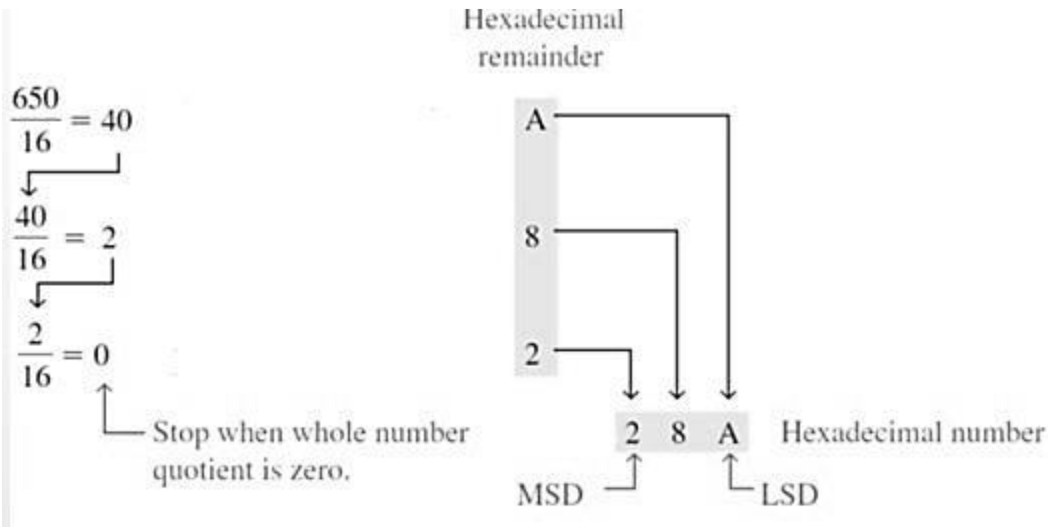
(b) A 8 5 → $101010000101 = 2^{11} + 2^9 + 2^7 + 2^2 + 2^0 = 2048 + 512 + 128 + 4 + 1 = \mathbf{2693_{10}}$

1. **Method Two:** Multiply the decimal value of each hexadecimal digit by its weight and then take the sum of these products. For a **4-digit** hexadecimal number, the weights are:

   **$B2F8_{16}$ = (B × 4096) + (2 × 256) + (F × 16) + (8 × 1) = (11 × 4096) + (2 × 256) + (15 × 16) + (8 × 1) = 45,056 + 512 + 240 + 8 = 45,81610**

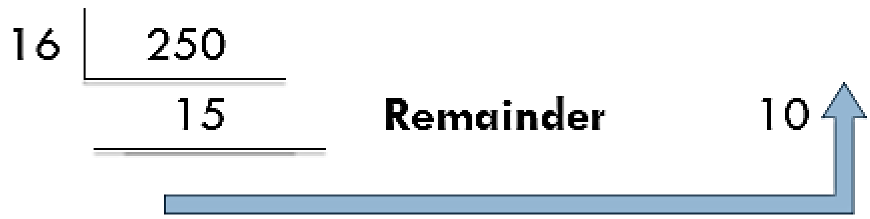
### C. Decimal-to-Hexadecimal Conversion (Repeated Division by 16)

Repeated **division by 16** will produce the equivalent hexadecimal number, formed by the remainders of the divisions. The **first remainder** produced is the least significant digit (**LSD**)




**Example : 250**

2 5 0




$$250_{10} = 15\ \ 10_{16}\ ?$$
$$= FA_{16}$$

## 4. Octal Numbers

Like the hexadecimal number system, the octal number system provides a convenient way to express binary numbers and codes. However, it is used less frequently than hexadecimal in conjunction with computers and microprocessors to express binary quantities for input and output purposes.

The octal number system is composed of **eight digits**, which are

**0, 1, 2, 3, 4, 5, 6, 7**

To count above 7, begin another column and start over:

**10, 11, 12, 13, 14, 15, 16, 17,**

**20, 21, …**

## 4.1 Number Base Conversion

### 1. Octal-to-Decimal Conversion

Since the octal number system has a base of eight, each successive digit position is an increasing power of eight, beginning in the right-most column with $8^0$. The evaluation of an octal number in terms of its decimal equivalent is accomplished by multiplying each digit by its weight and summing the products, as illustrated here for $2374_8$

**Weight: $8^3$ $8^2$ $8^1$ $8^0$**

$2374_8 = (2 \times 8^3) + (3 \times 8^2) + (7 \times 8^1) + (4 \times 8^0)$

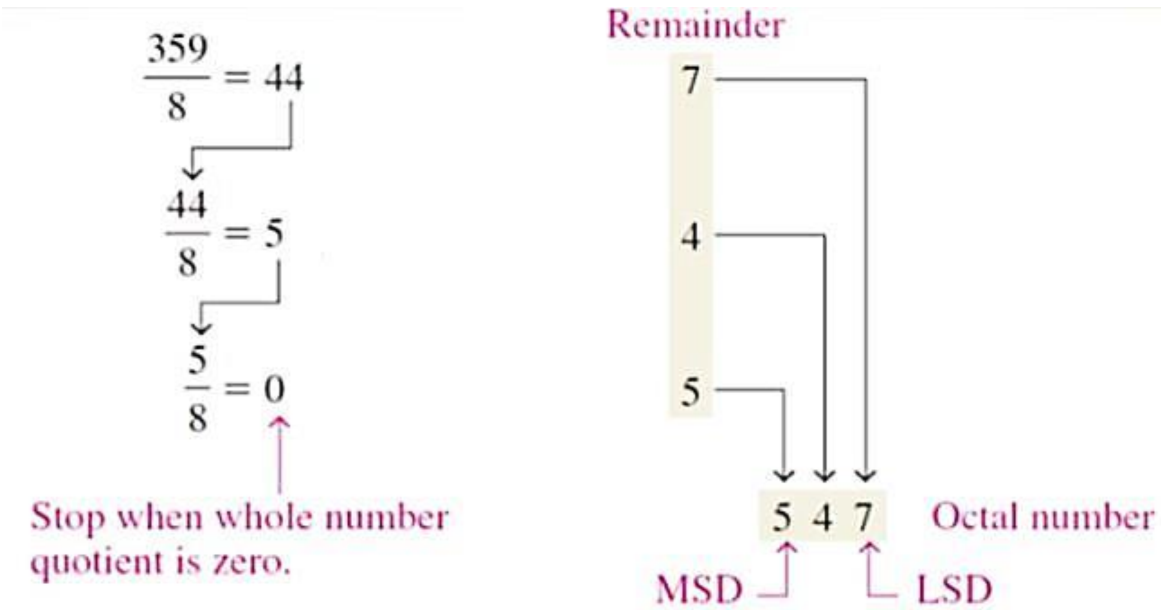
$= (2 \times 512) + (3 \times 64) + (7 \times 8) + (4 \times 1)$

$= 1024 + 192 + 56 + 4 = 127610$

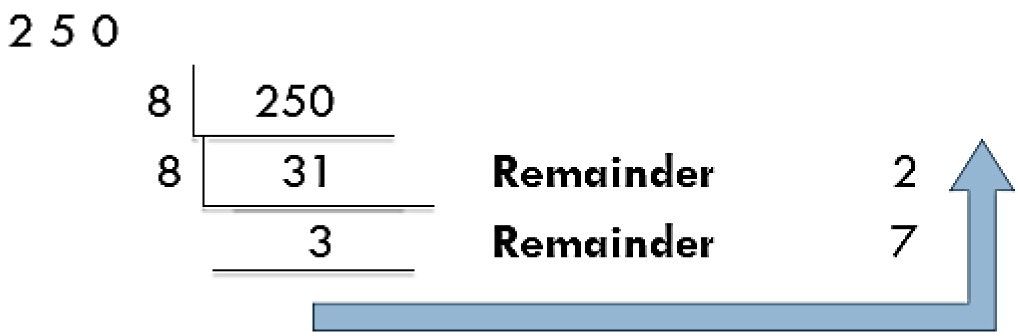
010 011 111 100

### 2. Decimal-to-Octal Conversion (Repeated Divisionby-8 Method)

Each successive division by 8 yields a remainder that becomes a digit in the equivalent octal number. The **first remainder** generated is the least significant digit (LSD).




**Example : 250**




$$250_{10} = 372_8$$

### 3. Octal-to-Binary Conversion

Because each octal digit can be represented by a **3-bit** binary number, it is very easy to convert from octal to binary. **Each octal digit is represented by three bits** as shown in **Table below**

Octal/binary conversion.

| Octal Digit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Binary | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

(a) 1 3 → 001011 (b) 2 5 → 010101 (c) 1 4 0 → 001100000 (d) 7 5 2 6 → 111101010110

## 4. Binary-to-Octal Conversion

**Start** with the **right-most group of three bits** and, moving from right to left, convert each **3-bit group** to the equivalent octal digit. If there are **not three** bits available for the left-most group, **add** either one or two zeros to make a complete group. These leading zeros do not affect the value of the binary number.

(a) 110101 → 6 5 = $65_8$

(b) 101111001 → 5 7 1 = $571_8$

(c) 100110011010 → 4 6 3 2 = $4632_8$

(d) 011010000100 → 3 2 0 4 = $3204_8$

**Table 2: Number system conversion**

| Decimal Number | Hexadecimal Number | Binary Number |
|---|---|---|
| 0 | 0 | 0000 |
| 1 | 1 | 0001 |
| 2 | 2 | 0010 |
| 3 | 3 | 0011 |
| 4 | 4 | 0100 |
| 5 | 5 | 0101 |
| 6 | 6 | 0110 |
| 7 | 7 | 0111 |
| 8 | 8 | 1000 |
| 9 | 9 | 1001 |
| 10 | A | 1010 |
| 11 | B | 1011 |
| 12 | C | 1100 |
| 13 | D | 1101 |
| 14 | E | 1110 |
| 15 | F | 1111 |

## HOME WORK

1. Convert the following binary numbers to hexadecimal:
   (a) 10110011 (b) 110011101000
2. Convert the following hexadecimal numbers to binary:
   (a) $57_{16}$ (b) $3A5_{16}$ (c) $F80B_{16}$
3. Convert $9B30_{16}$ to decimal.
4. Convert the decimal number 573 to hexadecimal.

## Binary Arithmetic العمليات الحسابية فى النظام الثنائى

### Binary Addition الجمع الثنائي

**Binary Addition**

The rules for binary addition are

| | |
|---|---|
| 0 + 0 = 0 | Sum = 0, carry = 0 |
| 0 + 1 = 0 | Sum = 1, carry = 0 |
| 1 + 0 = 0 | Sum = 1, carry = 0 |
| 1 + 1 = 10 | Sum = 0, carry = 1 |

When an input carry = 1 due to a previous result, the rules are

| | |
|---|---|
| 1 + 0 + 0 = 01 | Sum = 1, carry = 0 |
| 1 + 0 + 1 = 10 | Sum = 0, carry = 1 |
| 1 + 1 + 0 = 10 | Sum = 0, carry = 1 |
| 1 + 1 + 1 = 11 | Sum = 1, carry = 1 |

***EXAMPLE***

Add the binary numbers 00111 and 10101 and show the equivalent decimal addition.

```
 0111
 00111     7
 10101    21
------   ---
 11100  = 28
```

**Example 1**: add the binary no. 011, 110

**الحل:** نرتب الأعداد الثنائية بحيث تظهر في صورة أعمدة أو خانات واضحة كما يلي:

```
                1   1
      6             1   1   0
    + 3         +   0   1   1
   ----        ---------------
(عشري) 9        1   0   0   1
```

## Binary Subtraction

The rules for binary subtraction are

$0 - 0 = 0$
$1 - 1 = 0$
$1 - 0 = 1$
$10 - 1 = 1$,  0-1 with a borrow of 1

Subtract the binary number 00111 from 10101 and show the equivalent decimal subtraction.

```
 111
10101    21
00111     7
-----    --
01110 =  14
```

هناك طريقتان لإجراء عملية الطرح وهما :

1- الطريقة المباشرة أو ما يطلق عليه بالطريقة الحسابية.

2- الطريقة المتممة.

وسنكتفي هنا بشرح الطريقة المباشرة، وسوف نتناول الطريقة المتممة بالتفصيل فيما بعد. لإجراء الطرح بالطريقة المباشرة (الحسابية) يجب معرفة القواعد الأساسية لهذه العملية مع ملاحظة أن المقدار المطروح منه على اليسار والمقدار المطروح على اليمين:

$0 - 0 = 0$
$1 - 0 = 1$
$1 - 1 = 0$
$0 - 1 = 1$ ⟵ تكون النتيجة (1) واستلفنا (1)

ويمكن تلخيص عملية الطرح في الطريقة المباشرة كما يلي :

- رتب الأرقام تحت بعضها بحيث تظهر في صورة أعمدة أو خانات واضحة.
- ابدأ من الخانة الأولى على اليمين متجهاً إلى اليسار متبعاً القواعد التالية في الطرح:
  - عند طرح (0) من (0) أو (1) من (1) نضع في الناتج (0).
  - عند طرح (0) من (1) نضع الناتج (1).
  - عند طرح (1) من (0) نضع في الناتج (1) ثم نغير كل (0) من الخانات التالية (في المطروح منه) إلى (1) حتى نصل إلى أقرب (1) فنغيره إلى (0).
  - أكمل بعد ذلك علمية الطرح باستخدام القواعد السابقة.

**Ex.1** Subtract **011 from 101 :**

| | | | | |
|---|---|---|---|---|
| المطروح منه | 1 | 0 | ~~1~~ | |
| المطروح | 1 | 1 | 0 – | استلفنا (1) من العمود الذي يليه فأصبحت الخانة تحتوي على (10) وبطرح (1) منها يصبح الناتج (1) |
| | 0 | 1 | 0 | |

## <u>One's and Two's Complements of Binary Numbers</u>

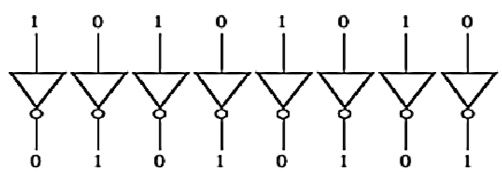
**<u>المتمم الاحادى والثنائى للأرقام الثنائية</u>**

### Finding the 1's Complement

**The 1's complement of a binary number is found by changing all 1s to 0s and all 0s to 1s, as illustrated below:**

| | |
|---|---|
| 1 0 1 1 0 0 1 0 | Binary number |
| ↓↓↓↓↓↓↓↓ | |
| 0 1 0 0 1 1 0 1 | 1's complement |

**The simplest way to obtain the 1's complement of a binary number with a digital circuit is to use parallel inverters (NOT circuits), as shown in Figure 2–2 for an 8-bit binary number.**




إن أهمية المتممين الأحادي والثنائي يكمن في سماحهما لنا بتمثيل الأعداد الثنائية السالبة. والمتمم الثنائي هو الأكثر شيوعاً واستخداماً في أجهزة الحاسوب للتعامل مع الأعداد السالبة. وللحصول على المتمم الأحادي لأي عدد ثنائي فإننا ببساطة نقوم بتغيير كل (1) إلى (0) ونغير كل (0) إلى (1) في العدد الثنائي كما يلي:

| | |
|---|---|
| العدد الثنائي ⟸ | 1 0 1 1 0 0 1 1 |
| | ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ |
| المتمم الأحادي ⟸ | 0 1 0 0 1 1 0 0 |

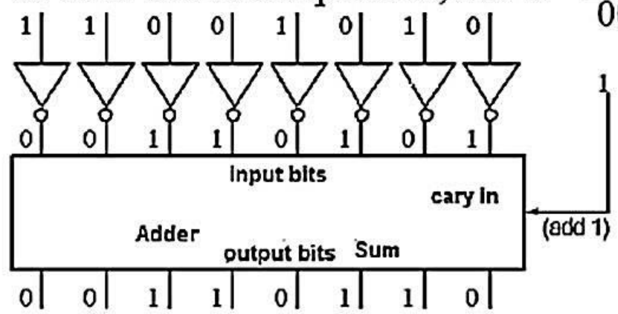
أما المتمم الثنائي للعدد الثنائي فإنه يمكن ايجاده بطريقتين كما يلي:

## 2 complement

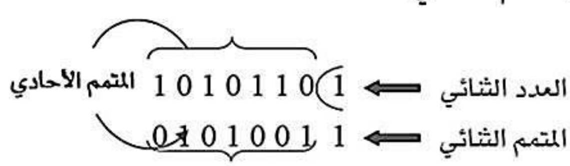
The 2's complement of a binary number is found by adding 1 to the LSB of the 1's complement.

Recall that the 1's complement of 11001010 is

| | | |
|---|---|---|
| | 00110101 | (1's complement) |
| To form the 2's complement, add 1: | +1 | |
| | 00110110 | (2's complement) |

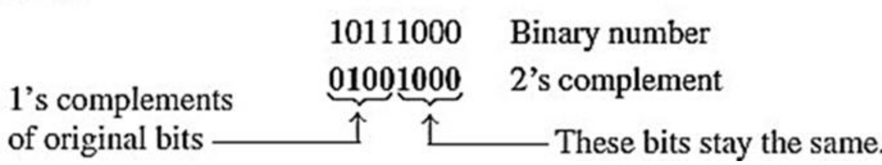



**الطريقة الثانية:** نقوم بالنظر للخانة الثنائية ذات القيمة الدنيا ($LSB$) من أقصى اليمين للعدد الثنائي فإن كانت تساوي (0) نقوم بكتابته ونستمر في ذلك وبمجرد أن نقابل أول خانة ثنائية تساوي واحداً عند ذلك نقوم بكتابة الواحد الذي قابلناه ثم بعد ذلك نقوم بقلب الصفر واحد اوالواحد صفراً وهكذا إلى أن ننتهي من كتابة العدد (وفي حال قابلنا أول واحد في الخانة الثنائية ذات القيمة الدنيا فإننا نقوم بكتابته ثم نتبع الطريقة السابقة بقلب الصفر إلى واحد والواحد إلى صفر) ومثال على ذلك، نفترض أننا نريد تحويل العدد الثنائي $(10101101)_2$ إلى المتمم الثنائي:




Find the 2's complement of 10111000 using the alternative method.

**Solution**




**Home Work**

**1-) Determine the weight of 1 in the number 10000**

**2-) Convert the binary number 1011101.011 to decimal.**

## عملية الطرح في النظام الثنائي باستخدام المتممات

### الطرح باستخدام المتمم الاحادى

لطرح عددين ثنائيين باستخدام المتمم الأحادي نتبع الخطوات التالية :

1. إكمال مراتب العدد الأقل عددا بالمراتب (المطروح أو المطروح منه ) .
2. إيجاد المتمم الأحادي للعدد المطروح .
3. جمع المتمم الأحادي للمطروح مع المطروح منه .
4. نلاحظ نتيجة الجمع للخطوة 3 وكما يلي :

أ. إذا كان هنالك واحد ظاهر في المرتبة الإضـــافية ، فنقوم بجمعه مع بقية العدد والناتج من عملية الجمع هو ناتج الطرح ويكون موجب .

ب. إذا لم يظهر واحد في المرتبة الإضـــافية ( وهو دلالة إن ناتج الطرح ســـالب ) ويكون ناتج الطرح بأخذ المتمم الأحادي لناتج الجمع للخطوة 3 ويكون ناتج العملية هو ناتج الطرح ويكون سالب.

**مثال (1)**

اطرح العدد $(110)_2$ من العدد $(1010)_2$ باستخدام طريقة المتمم الأحادي :

| | | |
|---|---|---|
| المطروح منه | 1010 | |
| المطروح | 110 − | |
| تكملة مراتب المطروح | 0110 | الخطوة 1 |
| المتمم الأحادي للمطروح | 1001 | الخطوة 2 |
| المتمم الأحادي للمطروح | 1001 | الخطوة 3 |
| المطروح منه | 1010 + | |
| المرتبة الإضافية ← | 10011 | الخطوة 4 |
| | 1 + | |
| ناتج الطرح ← | 0100 | |

**مثال (2)**

اطرح العدد $(10101)_2$ من العدد $(1011)_2$ باستخدام المتمم الأحادي :

| | |
|---|---|
| المطروح منه | 01011 |
| المطروح | 10101 − |
| المتمم الأحادي للمطروح | 01010 |
| المطروح منه | 01011 + |
| المرتبة الإضافية خالية إذن النتيجة سالبة ← | ?10101 |
| ناتج الطرح | 01010 − |

## Binary Codes

The electronic digital systems like computers, microprocessors etc., are required to process data which may include numbers, alphabets or special characters. The binary system of representation is the most extensively used one in digital systems i.e, digital data is represented, stored and processed as group of binary digits (bits). Hence the numerals, alphabets, special characters and control functions are to be converted into binary format. The process of conversion into binary format is known as binary coding. Several binary codes have developed over the years. Some of them are discussed in this section.

1. Binary coded decimal (BCD).
2. Gray code.
3. ASCII code

### 1- Binary Coded Decimal (BCD)

Internally, ***digital computers operate on binary numbers***. When interfacing to humans, digital processors, e.g. pocket calculators, communication is decimal-based.

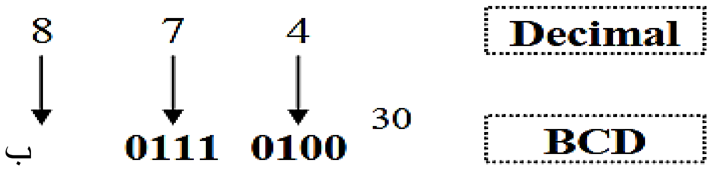
Input is done in decimal then converted to binary for internal processing. For output, the result has to be converted from its internal binary representation to a decimal form.

One commonly used code is the *Binary Coded Decimal* (BCD) code which corresponds to the first 10 binary representations of the decimal digits 0-9. The BCD code requires 4 bits to represent the 10 decimal digits. Since 4 bits may have up to 16 different binary combinations, a total of 6 combinations will be unused

Table (1)

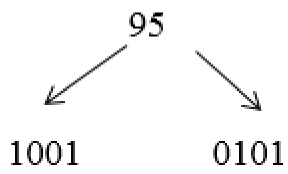
| Decimal Digit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| BCD Code | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 0100 | 1001 |

To illustrate the BCD code, take a decimal number such as 874. Each digit is changed to its binary equivalent as follows:




**Example**: Convert $(95)_{10}$ into BCD code .

**Solution:**




## 2- Gray Code

The gray code is un-weighted and is not an arithmetic code; that is, there are no specific weights assigned to the bit positions. The important feature of the Gray code is that it exhibits only a single bit change from one code number to the next.

Table (2) is a listing of the four bit gray code for decimal numbers 0 through 15. Notice the single bit change between successive gray code numbers. For instance, in going from decimal 3 to decimal 4, the gray code changes from 0010 to 0110, while the binary code changes from 0011 to 0100, a change of three bits. The only bit change is in the third bit from the right in the gray code; the other remain the same.

***Table (2)***

| Decimal | Binary | Gray | Decimal | Binary | Gray |
|---|---|---|---|---|---|
| 0 | 0000 | 0000 | 8 | 1000 | 1100 |
| 1 | 0001 | 0001 | 9 | 1001 | 1101 |
| 2 | 0010 | 0011 | 10 | 1010 | 1111 |
| 3 | 0011 | 0010 | 11 | 1011 | 1110 |
| 4 | 0100 | 0110 | 12 | 1100 | 1010 |
| 5 | 0101 | 0111 | 13 | 1101 | 1011 |
| 6 | 0110 | 0101 | 14 | 1110 | 1001 |
| 7 | 0111 | 0100 | 15 | 1111 | 1000 |

### Binary Number to Gray Code Conversion:

The procedures of conversion from binary to gray code are:

1. put down the MSB
2. start from the MSB, adding without carry each two adjacent bits

**Example**: convert the (10110)2 into gray code.

Solution:

$$\begin{array}{cccccccccc} 1 & -+\rightarrow & 0 & -+\rightarrow & 1 & -+\rightarrow & 1 & -+\rightarrow & 0 & \text{Binary} \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & \\ 1 & & 1 & & 1 & & 0 & & 1 & \text{Gray} \end{array}$$

***Example:*** convert the binary number 10110 to Gray code.

**Step 1:** the left-most Gray code digit is the same as the left-most binary code bit.

$$\begin{array}{ll} 10110 & \text{(Binary)} \\ \downarrow & \\ 1 & \text{(Gray)} \end{array}$$

**Step 2:** add the left-most binary code bit to the adjacent one:

$$\begin{array}{ll} \mathbf{1+0}\ 110 & \text{(Binary)} \\ \quad\ \downarrow & \\ 1\ \ 1 & \text{(Gray)} \end{array}$$

**Step 3:** add the next adjacent pair:

$$\begin{array}{ll} 1\ \mathbf{0+1}\ 10 & \text{(Binary)} \\ \qquad\ \downarrow & \\ 1\ \ 1\ \ 1 & \text{(Gray)} \end{array}$$

**Step 4:** add the next adjacent pair and discard the carry:

$$\begin{array}{ll} 1\ 0\ \mathbf{1+1}\ 0 & \text{(Binary)} \\ \qquad\quad\ \downarrow & \\ 1\ \ 1\ \ 1\ \ 0 & \text{(Gray)} \end{array}$$

**Step 5:** add the last adjacent pair:

1 0 1 **1 + 0** (Binary)

↓

1 1 1 0 1 (Gray)

***Hence the Gray Code is 11101***

## 2.2 Gray Code to Binary Number Conversion:

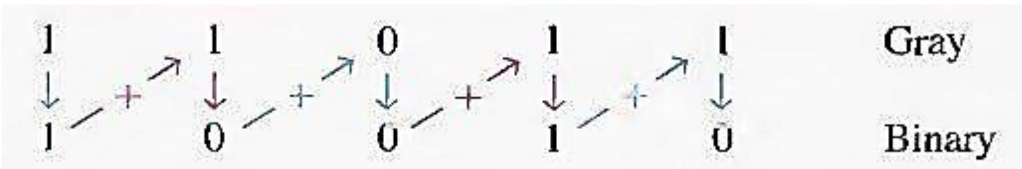
The procedure of conversion from gray code to binary are:

1- put down the MSB

2- start from the MSB adding without carry each result binary bit with the lower gray code bit

**Example:** convert the $(11011)_{gray}$ into binary.

**Solution:**




***Example:*** **convert the Gray code number 11011 to binary.**

**Step 1:** the left-most bits are the same.

11011 (Gray)

↓

1 (Binary)

**Step 2:** add the last binary code bit just generated to the gray code bit in the next position. Discard the carry.

1 **1** 011 (Gray)

+ ↓

**1** 0 (Binary)

**Step 3**: add the last binary code bit generated to the next Gray code bit.

1 1 **0** 11 (Gray)
+ ↓
1 **0** 0 (Binary)

**Step 4:** add the last binary code bit generated to the next Gray code bit.

1 1 0 **1** 1 (Gray)
+ ↓
1 0 **0** 1 (Binary)

**Step 5:** add the last binary code bit generated to the next Gray code bit. discard carry.

1 1 0 1 **1** (Gray)
+ ↓
1 0 0 **1** 0 (Binary)

**Hence the final binary number is 10010**

***Example:*** **(a)** Convert the binary number 11000110 to Gray-code. **(b)** Convert the Gray-code 10101111 to binary**.**

(a) Binary to Gray code:-

**1 + 1 + 0 + 0 + 0 + 1 + 1 + 0**
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
**1 0 1 0 0 1 0 1**

(a) Gray code to Binary

**1 0 1 0 1 1 1 1**
↓↗ ↓↗ ↓↗ ↓↗ ↓↗ ↓↗ ↓↗ ↓
**1 1 0 0 1 0 1 0**