Audio Text for the Lesson: Counting Systems

Part 1: Introduction (2 minutes)

Peace, mercy and blessings of Allah

Welcome everyone in this lesson

Allow me to introduce myself, my name is Abdullah Saleh Siddeq, a computer programmer, I would like to welcome you from Prince Sultan Bin Abdul Aziz Science and Technology Center (Scitech), supervised by King Fahd University of Petroleum and Minerals, Khobar, Kingdom of Saudi Arabia

I have an interesting lesson for you today. It is considered as the cornerstone of digital electronics and logical circuit designs such as calculators, electronic watches …etc.

I hope that you are ready to engage yourselves in a sea of numbers which are essential for computers, but before we start; we have the following things which are related to the computer:

This memory chip is for an old computer and its size is 256 MB, while this is a 4 GB flash memory, and this mobile phone has a memory of 2 GB.

Have you ever wondered why these numbers in particular? Why isn't there a memory chip with a size of 300 MB? Why haven't we heard about a memory size of 5 GB, for example? Try to find answers to these questions with your teacher assistance, and find a relationship among these numbers. I will come back to you shortly, God willing.

Part 2: The Decimal System (six minutes)

Welcome again.

As you have seen, these numbers are the results of the number 2 raised to the power of an integer, and to make this easier for you, I wrote the products of the number 2 raised to the power of the integer numbers from 0 to 10.

$2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 = 16$, $2^5 = 32$,
$2^6 = 64$, $2^7 = 128$, $2^8 = 256$, $2^9 = 512$, $2^{10} = 1024$.

I think these products are familiar to you in the computer field, for example, one megabyte equals to 1024 kilobytes and one kilobyte equals 1024 bytes, and one byte equals 8 bits. As you have seen, we cannot get 5 or 300 as a result of raising the number 2 to the power of an integer.

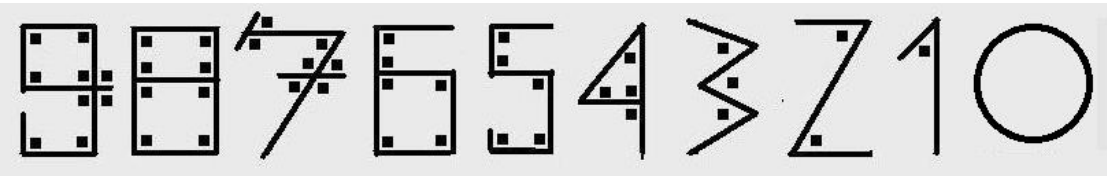
For reasons related to the internal design of these memories and the designs of the computer itself, which is based on the binary system, we find that sizes of memories always follow the number 2 raised to the power of an integer. The number 2 is the basis of the binary system which in turn is the base of computer design.

What is the binary system? What numbers are used in it? How can we deal with it?

Before I answer these questions, let me go back with you to the decimal system we used to deal with in our calculations and financial issues.

(The rest please, thank you)

Humans have ten fingers, so they have kept on using them a lot in counting and have depended a lot on their multiples. The Arab Muslims adopted the symbols of ten digits:



They have chosen a geometric figure for each digit with a number of angles equals to its value. Of course, soon these figures have been changed to the forms we know today, to make them easier to write and look better.

These ten symbols (digits) have allowed people to write any number easily no matter how big it is, for example, 1982 is made up of 1-9-8-2, where 2 represents 2 because it is in the ones digit, 8 represents 80 because it is in the in the tens digit, 9 is 900 because it is in the hundreds digit, and 1 represents 1000 because it is in the thousands digit. The Arabs and others always write numbers with the least digit value (Ones digit) on the right.

$1982 = (2 \times 10^0) + (8 \times 10^1) + (9 \times 10^2) + (1 \times 10^3)$

$= 2 + 80 + 900 + 1000 = 1982$

This is easy and clear and is applied to both binary and hexadecimal systems which we will learn about shortly.

How do you count from zero to a hundred?

Do not be surprised with this question. Let's look deeply into it. We have only ten symbols to represent any number, do not we?

When we get to the last symbol (9) we have to go back to (0) and increase the place value to its left by (1). 0-1 -... to... 9-10-11-… to… 19-20…to...-99 then 100. We are accustomed to this, and we see it when we watch the counter moves when we fill our car tank with fuel at fuel stations.

I think you are now ready to learn the binary system

Quite simply, it consists of only two symbols (Instead of 10 symbols) by which we can represent any number, and let's start counting by using this system: 0-1 - then what? There are no more symbols to use. There is no 2! What should we do? We will return to 0 and increase the left digit by 1.

0-1-10-11-100-101… My God! The digits of these numbers multiply rapidly, but it is alright, we have to deal with computers as an electronic machine that can recognize only two states, one with a specific voltage of electricity, which has been considered as the logical one, and a very little voltage of electricity, which has been considered as the logical zero.

Let us complete writing the numbers from 0 and up to 15

0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15.

As we did with the number 1982, try to find the value of the number 11011 in the decimal system. Work on this with your teacher taking into consideration that 10 is the basis for the decimal system, while 2 is the basis of the binary system. We will return shortly.

Part 3: Conversion to and from the Binary System (six minutes)(

11011

1×2^0

1×2^1

0×2^2

1×2^3

1×2^4

1-2-0-8-16

Total 27

So $(11011)_2$ in the binary system, corresponds to $(27)_{10}$ in the decimal system

Let's try another number that is slightly larger:

11001000

$0\times2^0 + 0\times2^1 + 0\times2^2 + 1\times2^3 + 0\times2^4 + 0\times2^5 + 1\times2^6 + 1\times2^7$

0 + 0 + 0 + 8 + 0 + 0 + 64 + 128 Total 200

What about trying an easier way? Watch this.

1	1	0	0	1	0	0	0
128	64	32	16	8	4	2	1

11001000

Write the value of each digit as in the table.

1-2-4-8-16-32-64-128.

We add the numbers under the ones and leave the rest 8 + 64 +128 = 200

Not only that, but we can convert a decimal number to a binary in this way as well, for example, 138:

We put 1 in the digit place with a value less than or equal to 138, which is in this case 128 and we subtract it from 138 to get 10 as a remainder, we keep subtracting with 8, and then with 2 to finish and we put zeros in the rest of the digit places to get the result: 10001010

Perhaps, I need to mention that there is only one set of those numbers totaling 138

But, there is a general method to convert from decimal to binary system which is based on repeated division of the number by 2 and to put the remainder, whether zero or one, from right to left next to each one to get the binary value until we get zero in the quotient, just as in the following example:

Number 366

366 divided by 2 equals 183 and the remainder is 0.

(Divided by 2) 91 and the remainder 1

(Divided by 2) 45 and the remainder 1

(Divided by 2)22 and the remainder 1

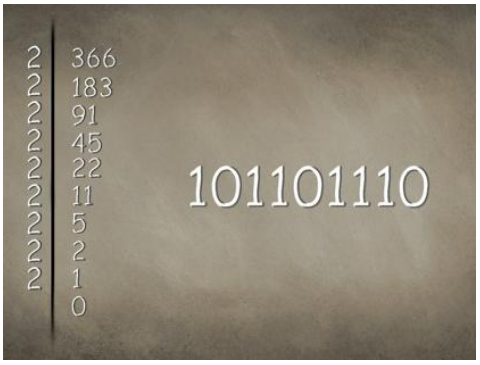
(Divided by 2)11 and the remainder 0

(Divided by 2)5 and the remainder 1

(Divided by 2)2 and the remainder 1

(Divided by 2)1 and the remainder 0

(Divided by 2)0 and the remainder 1



So, the result is 101101110

By using this method ,you can convert any number no matter how big, but if you try to solve this example using the previous method you will not be able to do so, because it is more than eight binary digits, or in other words, greater than 255.

But, programmers do not have enough patience to deal with numbers with too many digits, so they have created an easier system to work with; it is the hexadecimal system. This system consists of sixteen symbols which are: 0-1-2-3-4-5-6 -7-8-9-A-B-C-D-E-F.

Wait, these are not letters, but numbers, where A is a symbol of one digit and has the value equals to the number 10 in the decimal system, and similarly: B = 11, C = 12, D = 13, E = 14, and F = 15.

Now, I'll leave you for a few minutes to write hexadecimal numbers corresponding to the numbers from 0 to 100 in the decimal system; let one from each group use a pen to write –with his colleagues - these numbers. I am sure you can do it. Start please, and I'll be back shortly

Part 4: The hexadecimal system (four and a half minutes)

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

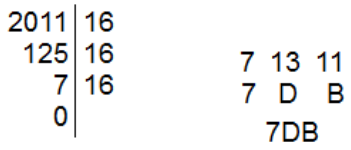
10-11-12-13-14-15-16-17-18-19-1A-1B-1C-1D-1E-1F

20-21-22-23-24-25-26-27-28-29-2A-2B-2C-2D-2E-2F

and so on…to

60-61-62-63-64

As you can see, it's easy, let's convert a decimal number to a hexadecimal number in a method like the one we used to convert to the binary system, repeat dividing the number by 16 until you get a zero quotient, and write the remainders from right to left by using the hexadecimal symbols.



Let us now convert a hexadecimal number to a decimal. I have a nice example for you; it is the number “FACE”. Do you like it?! It is not the English word meaning "face". It is a number!

$E \times 16^0 + C \times 16^1 + A \times 16^2 + F \times 16^3$

14 x1 + 12 x 16 + 10 x 256 + 15 x 4096

14 + 192 + 2560 + 1644 = 64206

In order to check our answer, write this number by using a calculator and convert it to a hexadecimal, you will find the number “FACE” written on its screen.

But, how programmers and other specialists in the information technology field benefited from the hexadecimal system to facilitate working with the binary system?

The basic storage unit in the computer as you know is the byte, which consists of eight digits of ones and zeros where each one of them is called bit, if we divide the byte into two parts and each of them consists of 4 bits, we get what we call nibble. Each nibble can be represented by one digit in the hexadecimal system, and that allows us to replace eight digits of zeros and ones by only two digits in the hexadecimal system, for example: $(F2)_{16} = (11110010)_2$, where

$(F)_{16} = (1111)_2$ and $(2)_{16} = (0010)_2$

And vice versa, we can convert from the hexadecimal system to the binary system. $(DC)_{16} = (11011100)_2$

The binary and hexadecimal numbers are just like decimal numbers since we can use addition, subtraction, multiplication and division, taking into consideration some cases where we face overflow, but this is not part of our lesson for today.

I'll leave you now with your teacher to work on some calculations on binary and hexadecimal systems using calculators. Meet you shortly

Part 5: Processes at the Bit Level (five minutes)

Hello, again. I hope you have enjoyed using the calculators, but I have found that when I add 655 to 598 in the hexadecimal system the addend is BED, but be careful this is not the English word which means "bed", but you should know very well that it is a number equal to 3053 in the decimal system.

If you tried -after this lesson- converting your mobile numbers to the hexadecimal system, you would find the digits become less and include some symbols (letters) Such as this

Now, let me tell you that the most important thing about binary numbers is that we can do operations other than, addition, subtraction, multiplication, and division with them. These operations are the logical operations and they are done at the bit level. There is a set of logical operations. We will talk today about the most important two of them: AND, & OR.

As we do with multiplication in the decimal system, we multiply two numbers and we get one number as a product. Similarly, multiplication of the logical operator AND produces one bit as a result. While the logical operator OR is like, but not identical to the addition of decimals, where you add two bits and get one bit as a result taking into consideration that 1 OR 1 = 1, as in the following tables

But, What is the real benefit from all these operations, symbols, digits, and systems?

In fact, when a computer modifies or creates any file of any type, it only writes these digits, and performs these operations on them, and for this reason, a computer is called a computer.

For example, we can change an image in a computer through these operations where we can remove all degrees of red. How? For example, if the color of a point on the image is yellow, and yellow as we know, is produced from red and green in the absence of blue, so when we remove the red color from the yellow color, the point becomes green.

Note that in the computer language, color is represented by six digits in the hexadecimal system, where each two digits represent a color: two for red, two for green, and two for blue, respectively.

The yellow point has the value: 00FFFF

If we apply the AND operation between this value and the value FFFF00

The result is 00FF00 which is green.

Let me explain this by using the binary system:

Yellow: 0000,0000,1111,1111,1111,1111

To delete Red: 1111,1111,1111,1111,0000,0000

By applying AND: 0000,0000,1111,1111,0000,0000

Which is: 00FF00

This is an application about that by using Visual Basic language. You can download it from BLOSSOMS website.

Programming languages differ with each other slightly when performing operations at the bit level; this is a sample problem from Visual Basic language. I hope that you are able to solve it with your teacher supervision.

If:

Byte1 = 10101100

&

Byte2 = 01011000

Find the value of Byte3 where:

Byte3 = (Byte1 And &HF0&) Or ((Byte2 And &HF0&) \ 16)

Part 6: Conclusion (three minutes)

Byte3 = (Byte1 And &HF0 &) Or ((Byte2 And &HF0 &) \ 16)

10101100 and

11110000

==========

10100000

01011000 and

11110000

==========

01010000

/16

=========

00000101

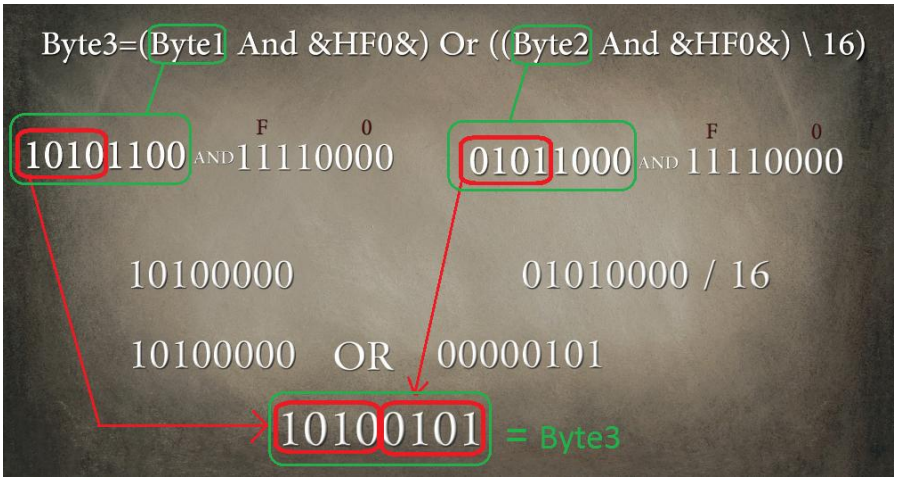
10100000 or

00000101

=========

10100101

The question we have solved is the most important part in the program "The Magic Picture" which you can find in one of the BLOSSOMS lessons titled "The Magic Picture: Hiding Data in Bitmaps" I encourage you to watch it and benefit from it.



It is my great pleasure to work with counting systems and their operations, and if you agree with me, I invite you to study it deeply and learn other operations that we have not covered

Such as:

NOT, NOR, NAND, XAND, XNOR

And it usage in building digital circuits and ciphering softwares

Finally, I would like to mention that you can e-mail me if you have any questions. It will be a pleasure to be in touch with you.

I leave under the auspices of Allah, peace, mercy and blessings of Allah

Part 7: (Teacher's Guide) (three minutes)

Dear, Teachers: Peace, mercy and blessings of Allah

Thank you for choosing this lesson about counting systems in computer.

The purpose of this lesson is to introduce students to the binary, and hexadecimal systems in a simplified way, and teach them conversion methods between them, and carrying out operations on the bit level by using these systems.

This lesson does not require additional knowledge other than what the students have learned in their previous math lessons, such as raising a number to a power and the standard math operations (+-÷x)

You will not need special materials for this lesson other than a calculator program found in Windows operating system or those available in some mobile phones or scientific calculators; you can also use letters and numbers from Toys to add a funny flavor to the lesson.

In the fourth activity, I hope that you start by writing numbers on the board, such as 5, 8 and 12, for example, then using the calculator to convert them to the binary system 101-1000-1100 and the hexadecimal 5-8-C, then performing addition, and subtraction within each system of the three systems and converting each result to other systems to help the students check performing addition and subtraction, and maybe it would be appropriate to try subtracting 8 from 5 so that the result is -3 in the decimal system while the result will be wrong in the binary and hexadecimal system and this is because of underflow.

In the fifth activity, I hope that you explain to your students that the symbols &H& are placed around the hexadecimal number in order to be recognized by the programming language as a hexadecimal number, and kindly explain that dividing a hexadecimal number by 16 is similar to

dividing a decimal number by 10, which means moving the digits in the hexadecimal number one digit to the right. (The number 16 is 2^4)

Finally, the number 10 in the joke I mentioned in the written teacher's guide is a binary number equal to 2, which means that there are two types of people; those who understand the binary system and those who do not understand it.

I hope that those in charge of this project and me have successfully provided a useful resource to the scientific library, and peace, mercy, and blessings of Allah.