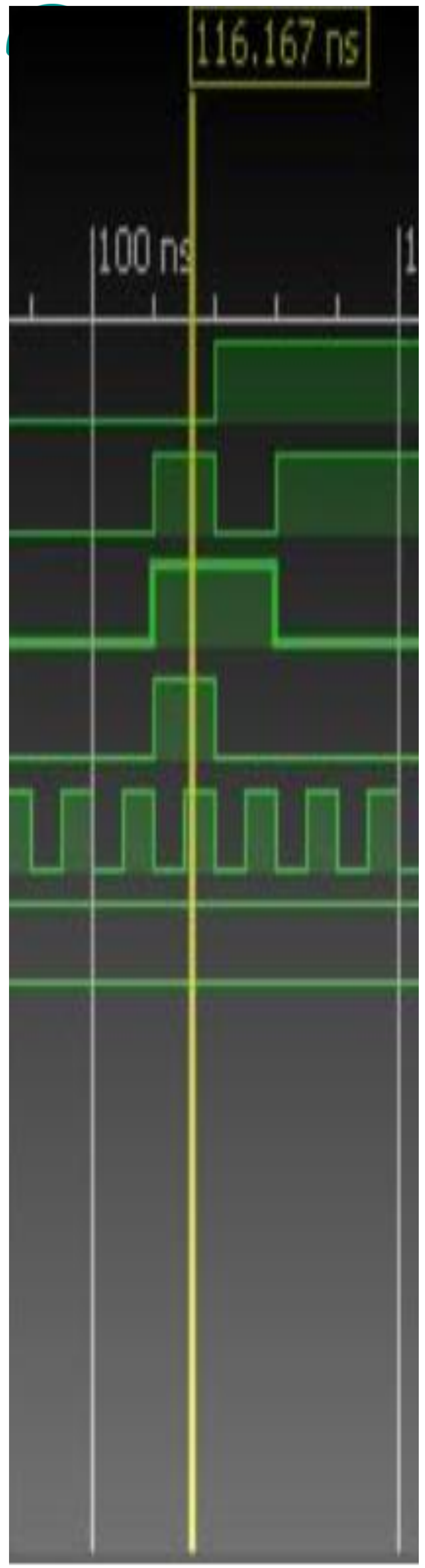
Number Systems and Arithmetic

MC. Martin González Pérez

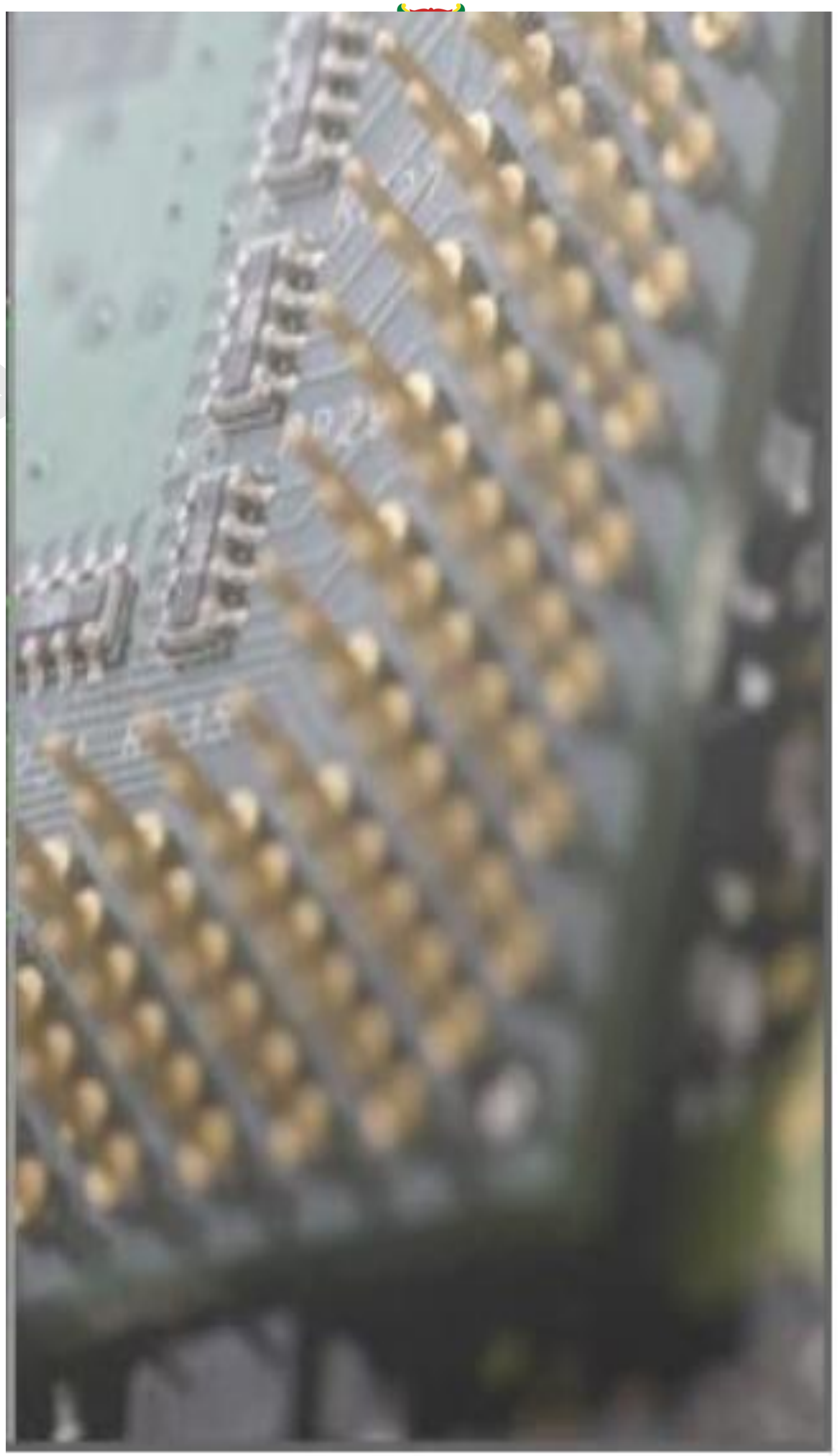
DIGITAL DESIGN

Agenda

- Number system
- Number system conversion
- Binary number system



Number systems



Number systems

A numerical system is a set of rules and symbols used to represent quantities and perform mathematical operations. Each system has its base, which determines the number of available digits and how their values are interpreted. Currently, decimal number system it is the most widely used system by humanity.

0 1 2 3 4

5 6 7 8 9

Decimal number system

The decimal number system works by using ten digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) to represent any quantity. This system is positional and base 10, which means that each position to the left increases its value in powers of ten.

position

$$325 = 3x10^{2} + 2x10^{1} + 5x10^{0}$$

symbol

base

Decimal number system : fractions

In the decimal number system, fractions are represented by using a decimal point to indicate the start of negative powers of 10. To the right of the decimal point, each digit represents a tenth (10^{-1}), then a hundredth (10^{-2}), a thousandth (10^{-3}), and so forth.

$$14.72 = 1x10^1 + 4x10^0 + 7x10^{-1} + 2x10^{-2}$$

$$\underbrace{\mathbf{10} \quad + \quad \mathbf{4}}_{\text{integers}} \quad + \quad \underbrace{\mathbf{0.7} \quad + \quad \mathbf{0.02}}_{\text{fractions}}$$

• Decimal point

Binary number system

Binary number system uses only two digits: 0 and 1. This system is positional and base-2, which means that each position to the left increases its value in powers of two.

$$1110 = 1x2^3 + 1x2^2 + 1x2^1 + 0x2^0$$

$$= \mathbf{8} + \mathbf{4} + \mathbf{2} + \mathbf{0}$$

Positive Powers of Two
(Whole Numbers)

2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
256	128	64	32	16	8	4	2	1

Digits on binary system are called bits.

Martin González Pérez
martin.perez@cinvestav.mx

Binary number system : fractions

Just like the decimal numbers, the binary system can represent fractions using a fixed point. Similarly to the decimal system, the digits to the right of the point are multiplied by the base raised to a negative power.

$$10.11 = 1x2^1 + 0x2^0 + 1x2^{-1} + 0x2^{-2}$$

$$= \mathbf{2} + \mathbf{0} + \mathbf{1/2} + \mathbf{1/4}$$

Positive Powers of Two (Whole Numbers)								
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
256	128	64	32	16	8	4	2	1

Negative Powers of Two (Fractional Number)					
2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
1/2 0.5	1/4 0.25	1/8 0.125	1/16 0.625	1/32 0.03125	1/64 0.015625

Hexadecimal number system

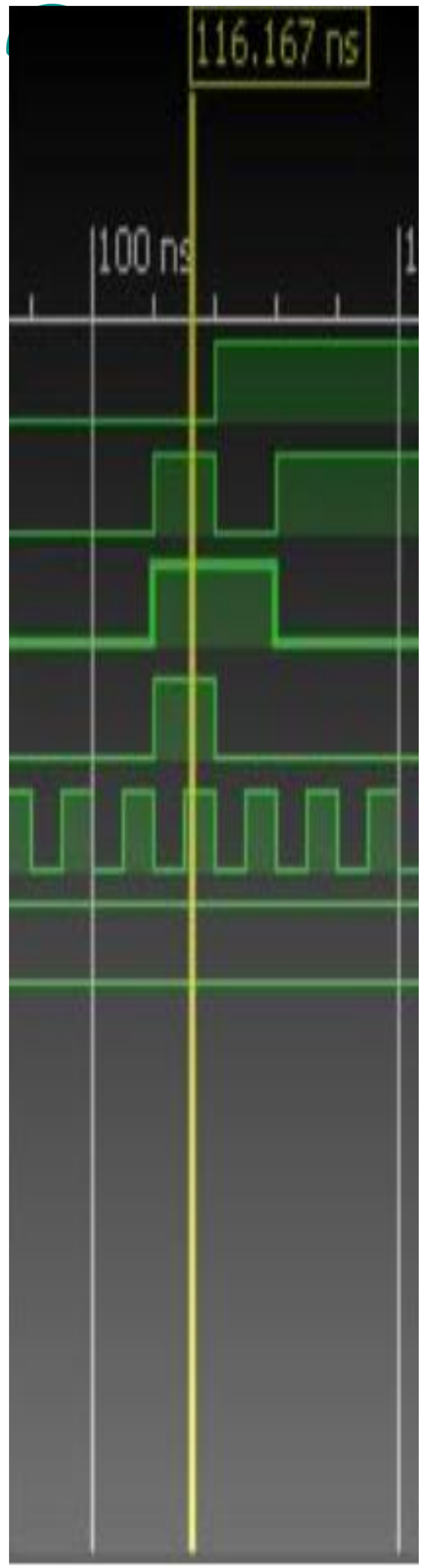
The hexadecimal numbers uses sixteen digits: from 0 to 9 and then the letters A, B, C, D, E, F to represent values from 10 to 15. Each position to the left increases its value in powers of 16.

$$3A = 3x16^1 + 10x16^0 = 48 + 10 = 58$$

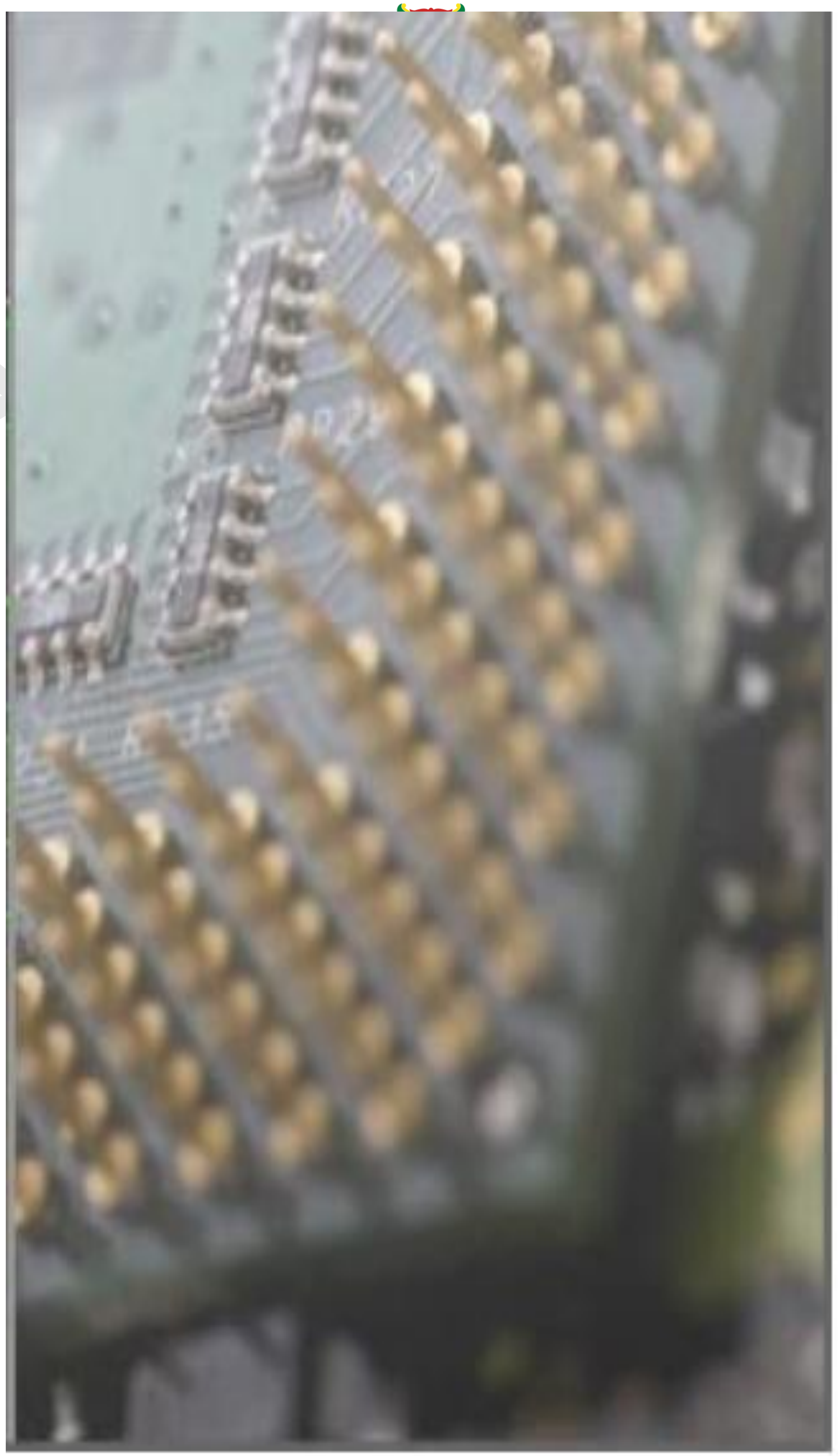
Hexadecimal number system

Since it consists of 16 digits, the hexadecimal system can be represented using 4 bits (a nibble). This is very convenient for handling information in digital systems, as working directly with binary numbers can be tedious and prone to errors.

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



Number systems conversion



Binary to decimal conversion

The decimal value of any binary number can be found by adding the weights of all bits that are 1 and discarding the weights of all bits that are 0.

- Exercise: Convert the binary number 0110_1101 to decimal

$$01101101 = 0x2^7 + 1x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$$

$$= 0 + 64 + 32 + 0 + 8 + 4 + 0 + 1 = \mathbf{109}$$

$\underbrace{7\,6\,5\,4\,3\,2\,1\,0}_{\text{position}}$

Binary to hexadecimal conversion

Converting a binary number to hexadecimal is a straightforward procedure. Simply break the binary number into 4-bit groups, starting at the right-most bit and replace each 4-bit group with the equivalent hexadecimal symbol.

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

- Convert the following binary numbers to hexadecimal:

a) 1100_1010_0101_0111 **b)0011_1111_0110_1001**

Solution:

1100_1010_0101_0111 → C A 5 7

0011_1111_0110_1001 → 3 F 6 9

Hexadecimal to binary conversion

To convert from a hexadecimal number to a binary number, reverse the process and replace each hexadecimal symbol with the appropriate four bits.

- Determine the binary numbers for the following hexadecimal numbers:

a) 10A4 **b)CF8E**

Solution:

1	0	A	4

0001_0000_1010_0100

C	F	8	E

1100_1111_1000_1110

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

Exercises

Convert form binary to hexadecimal:

a) 0011_0110 b) 0001_1110 c)1111_1010

Convert from hexadecimal to binary:

a) $7F_{16}$ b)80_{16} c)96_{16}

Convert from decimal to binary:

a) 57_{10} b)34_{10} c)7_{10}

Exercises

Convert form binary to hexadecimal:

a) 0011_0110 b) 0001_1110 c)1111_1010

36 1E FA

Convert from hexadecimal to binary:

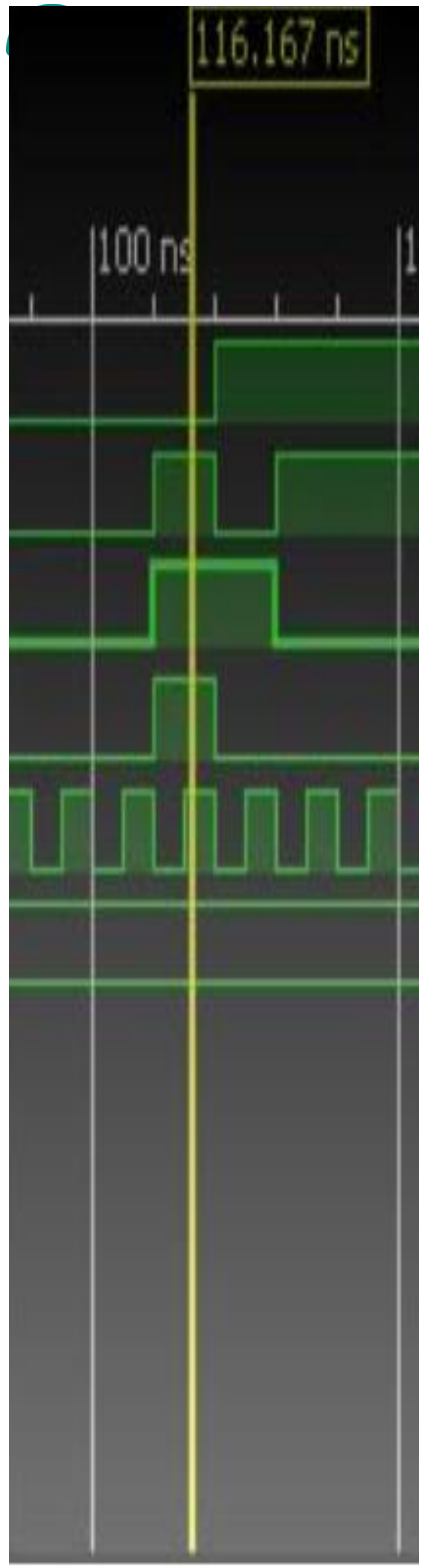
a) $7F_{16}$ b)80_{16} c)96_{16}

0111_1111 1000_0000 1001_0110

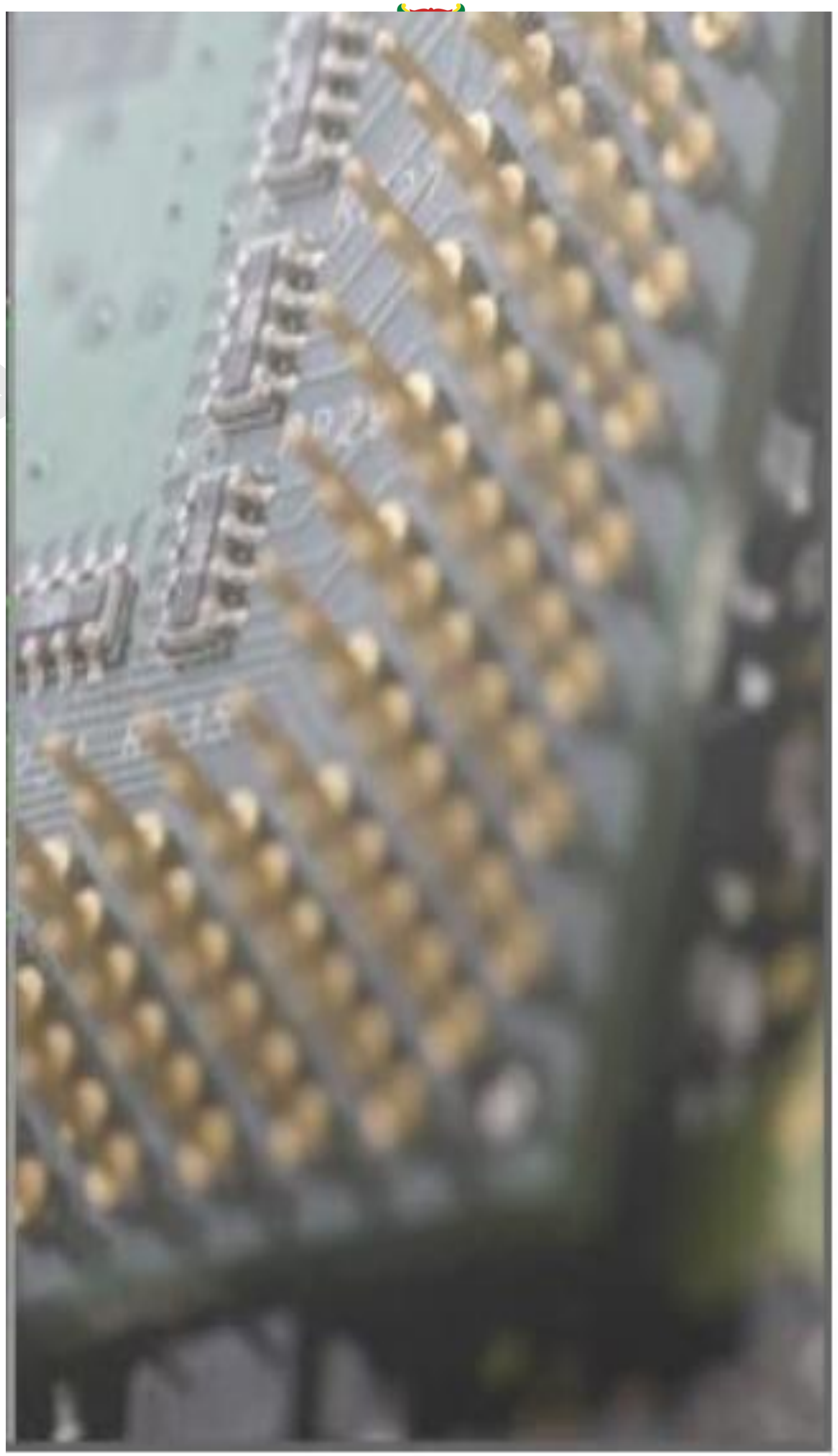
Convert from decimal to binary:

a) 57_{10} b)34_{10} c)7_{10}

111001 100010 111



Binary number systems



Bits, Bytes, Nibbles

Bit: The smallest unit of information in a digital system, which can take one of two values: 0 or 1.

Nibble (Half Byte): A group of 4 bits. It is a small data unit larger than a single bit but smaller than a Byte. A nibble can represent a single hexadecimal digit (0 to F).

Byte: A standard grouping of 8 bits. A Byte is commonly used as the basic unit for representing a character in many computer systems and serves as a standard measure for storage capacity.

$$\overbrace{1001\_\underbrace{0110}_{\text{Nibble}}}^{\text{Byte}}$$

$$\underbrace{\text{CE}}_{\substack{\text{Most}\\\text{significant}\\\text{Byte}}}\text{BFF9}\underbrace{\text{D7}}_{\substack{\text{Least}\\\text{significant}\\\text{Byte}}}$$

Numbers representation

- How many different values can a single bits represent? And **n** bits?

$$2^n$$

- Which is the maximum natural number that can be represented with a bit? And **n** bits?

$$2^n - 1$$

Addition

Binary addition is like decimal addition. As in decimal addition, if the sum of two numbers is greater than what fit in a single digit, we carry a 1 into the next column.

$$
\begin{array}{r}
{\color{red}11} \quad {\color{red}\longrightarrow \text{Carry}} \\
1011 \\
+\ 0011 \\
\hline
1110
\end{array}
$$

Overflow

Digital systems usually operate on a fixed number of digits. Addition is said to *overflow* if the result is too big to fit in the available digits. A 4-bit number, for example has the range [0,15], if a 4-bit addition *overflow* if the result exceed 15. *Overflow* can be detected by checking for a carry out of the most significant column.

$$
\begin{array}{r}
1\ \ 1 \\
1101 \\
+\ 0101 \\
\hline
10010
\end{array}
$$

overflow

What about negative numbers?

So far, we have considered only *unsigned* numbers that represent positive quantities. To represent negative numbers, we need a different numerical system. The two most widely employed are called **sign/magnitude** and **two's complement**.

Sign/Magnitude Numbers

Sign/magnitude numbers are intuitive appealing because they match our custom of writing negative numbers with a minus sign followed by the magnitude. A N-bits sign/magnitude number uses the N bit (Most significant bit) to represent the sign and the remaining N-1 bits as the magnitude. A sign bit equal to 0 indicates a positive number and a sign bit equal to 1 indicates a negative number.

What is the range for a N-bits sign/magnitude number?

$$[-2^{N-1}+1, 2^{N-1}-1]$$

Sign/Magnitude Numbers

A drawback of sign/magnitude number system is that ordinary addition doesn't work.

- Perform de addition of 3 and -3 using the sign/magnitude number system.

$$
\begin{array}{r}
1011 \\
+\ 0011 \\
\hline
1110
\end{array}
$$

Incorrect!

Martin González Pérez
martin.perez@cinvestav.mx

Two's complement

Two's complement numbers are identical to unsigned binary numbers except that the most significant bit position has a weight of -2^(N-1) instead of 2^(N-1).

What is the range for a N-bits sign/magnitude number?

$$[-2^{N-1}, 2^{N-1} - 1]$$

Two’s complement

- Represent -24 as 8-bit two’s complement number.

Step1: Represent the magnitude as unsigned number.

Step2: Invert the bits.

Step3: Add 1.

Unsigned binary representation		Invert bits		Add 1
0 0 0 1 1 0 0 0	→	1 1 1 0 0 1 1 1	→	1 1 1 0 1 0 0 0

$$-2^7 + 2^6 + 2^5 + 2^3 = -128 + 64 + 32 + 8 = -24$$

Martin González Pérez
martin.perez@cinvestav.mx

Two's complement (subtraction)

Two's complements numbers have the advantage that addition works properly for both positive and negative numbers.

A subtraction can be performed by taking the two's complement of the second number.

- Perform the subtraction 5-7 using 4-bit two's complement numbers.

$$
\begin{array}{rr} & 0101 \\ - & 0111 \\ \hline \end{array}
\Rightarrow
\begin{array}{rr} & 0101 \\ + & 1001 \\ \hline & 1110 \end{array}
\rightarrow -2^3 + 2^2 + 2^1 = -8 + 4 + 2 = -2
$$

Two’s complement (considerations)

- Perform the subtraction 5-3 using 4-bit two’s complement numbers.
- Perform the addition of 4 and 5 using 4-bit two’s complement numbers.
- Perform the addition of -4 and -5 using 4-bit two’s complement numbers

$$\begin{array}{r} 0101 \\ +\ 1101 \\ \hline 10010 \end{array}$$

correct

discarded

$$\begin{array}{r} 0101 \\ +\ 0100 \\ \hline 1001 \end{array}$$

incorrect

overflow

$$\begin{array}{r} 1011 \\ +\ 1100 \\ \hline 10111 \end{array}$$

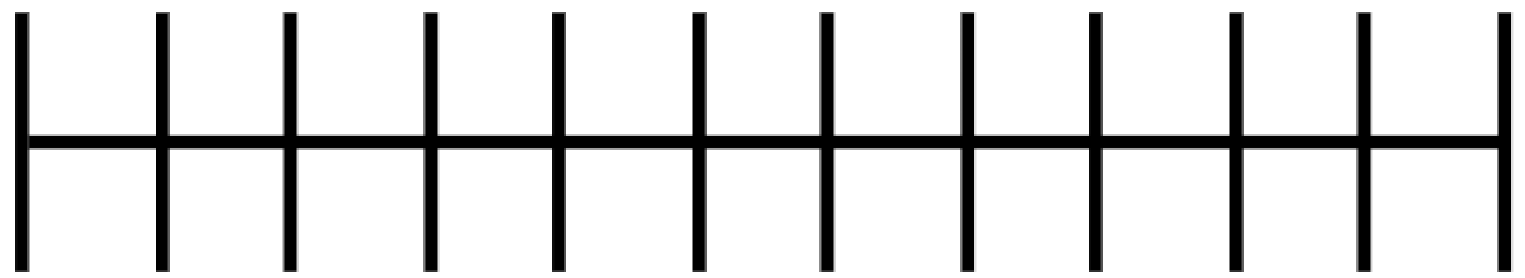
incorrect?

overflow

<u>It is important to consider the range of values that can be represented in the result in order to detect overflow. Overflow in above examples can be solved using 5 bits.</u>

Comparison of number systems

System	Range
Unsigned	$[0, 2^N - 1]$
Two's Complement	$[-2^{N-1}, 2^{N-1} - 1]$
Sign/Magnitude	$[-2^{N-1} + 1, 2^{N-1} - 1]$



	-4	-3	-2	-1	0	1	2	3	4	5	6	7
Unsigned					000	001	010	011	100	101	110	111
Two Complement	100	101	110	111	000	001	010	011				
Sign/Magnitude		111	110	101	100 000	001	010	011				

Exercises

What is the range of numbers that can be represented with 32 bits in unsigned integer format?

Represent the next numbers in sign/magnitude and two's complement formant. Use 8-bits.

- -124
- 200

Perform the subtraction (100-40) using two's complement.

Exercises

What is the range of numbers that can be represented with 32 bits in unsigned integer format?

4,294,967,295

Represent the next decimal numbers in sign/magnitude and two's complement formant. Use 8-bits.

- -124 SM- 1111_1100 2Comp- 1000_0100
- 200 is not possible

Perform the subtraction (100-40) using two's complement.

0011_1100