

Contents

P1. "Numeric & Alphanumeric Data "	2
P2: Explain, using examples, how different types of data can be converted and stored in computer systems.	7
P3: "Conversions of Numeric Data"	14
P4. Carry Out Boolean logic Operations.	24
P5. "Key Computer System Components"	33
P6: "Different Types of Memory"	44
P7. Explain how polling and interrupts are used to allow communication between processor and peripherals.	48
P8. RISC and CISC	49
P9. "Fetch-Execute Cycle"	51
M1. "Floating Point Numbers"	56
M2. "Roles Played by Different Memories"	59
M3. "Low-level Program"	62

P1. “Numeric & Alphanumeric Data “

(a) Sign and magnitude

In sign and magnitude, the left of the digit signifies the sign of the number. It is used commonly in computers. The sign and magnitude are integer when we use we have to identify whether it is positive or negative. If it is positive binary number, the sign will be 0 and if it is negative binary number the sign will be 1. And last, the remaining of the binary number are marked as the magnitude.

$$+7_{10} = 0111_2$$

$$-7_{10} = 1111_2$$

In the above, 111_2 is displayed as a magnitude and the red color is represented as a sign bit. In these sign bits, 0 is represented as (+) sign and 1 is represented as (–) sign.

(b) Two's Complement

Two's complement is the useful method of describing the signed integers on computers. In computer, the numbers are stored in registers where there is reserved designated number of bits for the storage of numbers in the binary form. Registers are come into various sizes. It is easy to change a negative integer with base 10 into binary form using two's complement method. There are two steps in this method they are one's and two's complement. At the one's complement, the number behaves like a negative and we have to reverse by changing zeroes to ones and ones to zero. At the two's complement, there should be needed to add 1 to the result of one's complement. When these stages are successfully completed, we have to sum with positive digit and also with the result of the two's complement. If there is an extra number in the result, it will be marked as abrogation. The examples of the two's complements are as shown in below.

$$2_{10} + (-6_{10}) = 0010_2 + (-0110_2)$$

$$0010_2$$

$$0110_2$$

$$1001_2$$

$$+1$$


$$1010_2$$


$$+ 0010_2$$

$$1100_2$$

$$0011_2$$

$$+1$$

$$0100$$
 One's complement

 Two's complement

(b) Two's Complement

$$9_{10} + (-6_{10}) = 1001_2 + (-0110_2)$$

$$1001_2$$

$$0110_2$$

$$1001_2$$

$$+1$$

$$1010_2$$

$$1001_2$$

$$\neq 0011$$

(c) Floating Point

In the programming fields, a floating point is used to store floating point number values. It is also famous in maths and programming fields as a decimal number. Floating numbers are stored in a computer as three fields which are the sign, exponent and the significand or otherwise it means mantissa respectively. As an example the floating number in 32 bits divided into fields which are 1 bit for the sign, 8 bits for the exponent and 23 bits for the mantissa.

$$34.25_{10} = 100010.01_2$$

$$= 1.0001001 \times 2^5$$

2	34	0
2	17	1
2	8	0
2	4	0
2	2	0
2	1	1
0		

Multiply by 2 (Calculate the fraction part)

.25x2 0

.5x2 1

0

$$34_{10} = 100010_2$$

$$.25 = .01_2$$

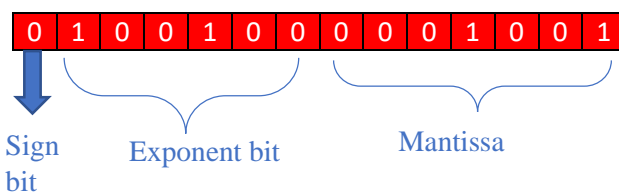
$$\text{Floating point} = 100010.01_2 = 1.0001001 \times 2^5$$

$$\text{Exponent} = 5$$

$$\text{Exponent bit} = 5 + 31 = 36 = 100100_2$$

$$\text{Mantissa} = 0001001$$

$$\text{Sign bit} = 0$$



$$42.625_{10} = 101010.101_2$$

$$= 1.01010101 \times 2^5$$

2	42	0
2	21	1
2	10	0
2	5	1
2	2	0
2	1	1
0		

Multiply by 2 (Calculate the fraction part)	
.625x2	1
.25x2	0
.5x2	1
0	

$$42_{10} = 101010$$

$$.625_{10} = .101_2$$

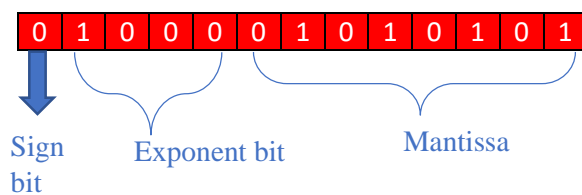
$$\text{Floating point} = 101010.101_2 = 1.01010101 \times 2^5$$

$$\text{Exponent} = 5$$

$$\text{Exponent bit} = 5 + 3 = 8 = 1000_2$$

$$\text{Mantissa} = 01010101$$

$$\text{Sign bit} = 0$$



BCD

BCD means Binary Coded Decimal and it is a type of binary code. It is easy to represent to represent the decimal values. BCD is an easy way to describe each of the decimal digits with a binary code. A decimal digit in BCD is same as its equivalent binary number only when the number is between 0 and 9. In this BCD, 4 bits represent each decimal digit. The examples are as shown in below

$$445_{10} = 0100\ 0100\ 0101\ (\text{BCD})$$

$$7542_{10} = 0111\ 0101\ 0100\ 0010\ (\text{BCD})$$

ASCII

ASCII (American Standard Code for information Interchange) which is a numeric code and used to represent the characters. Log time ago, different computer manufacturing companies use the binary system in their own way practically. Nowadays, all of the computer manufactures agreed to use ASCII. And then, all of the computer brands can use with one binary system. As I observed that if we type the text in computer it cannot work as directly and need to change binary. Alternatively, ASCII is an 8-bit code and it uses eight bits to describe a letter. In other way, eight bits can be called a byte. A binary code that is with eight digits like $1011\ 1101_2$ can be stored in one byte of the computer.

Here is the example for the ASCII

$$G=47_{16} = 01000111_2$$

$$r = 72_{16} = 01110010_2$$

$$e = 65_{16} = 01100101_2$$

$$y = 79_{16} = 01111001_2$$

So, the ASCII for the Grey is $01000111_2\ 01110010_2\ 01100101_2\ 01111001_2$

$$P=50_{16} = 01010000_2$$

$$i=69_{16} = 01101001_2$$

$$c=63_{16} = 01100011_2$$

$$o=6F_{16} = 01101111_2$$

So, the ASCII for the Pico is $01010000_2\ 01101001_2\ 01100011_2\ 01101111_2$

P2: Explain, using examples, how different types of data can be converted and stored in computer systems.

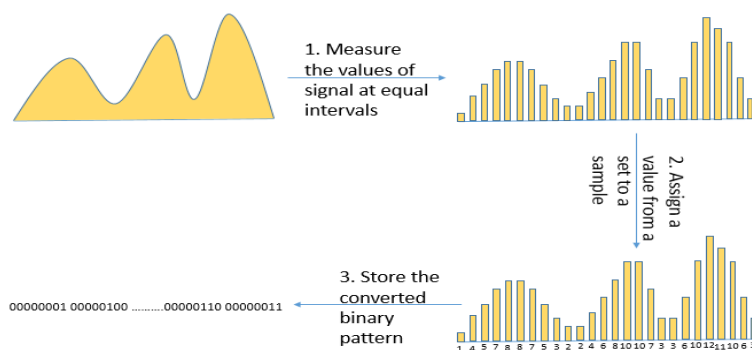
1. (a) Text

When we press any key on a keyboard, it will be converted into a binary number and that can be done by the computer by way of using ASCII value and then will appear on a computer screen what kind of letter that we type. As an example, when we type "BRAVE" it will convert to binary number by using ASCII value.

Text	B	R	A	V	E
Binary number	01000010	01010010	01000001	01010110	01000101
Decimal	66	82	65	86	69

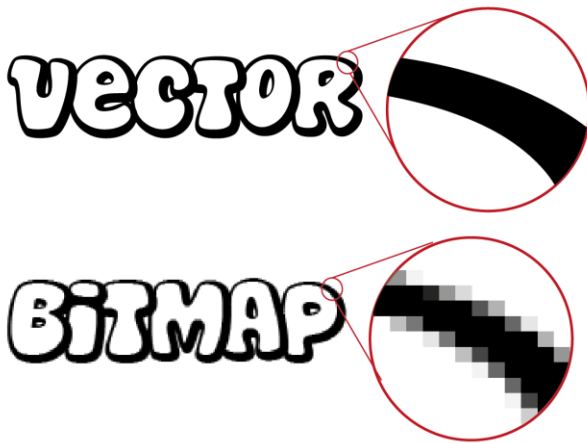
(b) Audio

In the audio, it also must be changed into binary. The sound is already recorded and then converted into the digital form. The sound samples are signified as an amplitude to the digital signal. This amplitude is maintained as an integer and stored as a binary number.



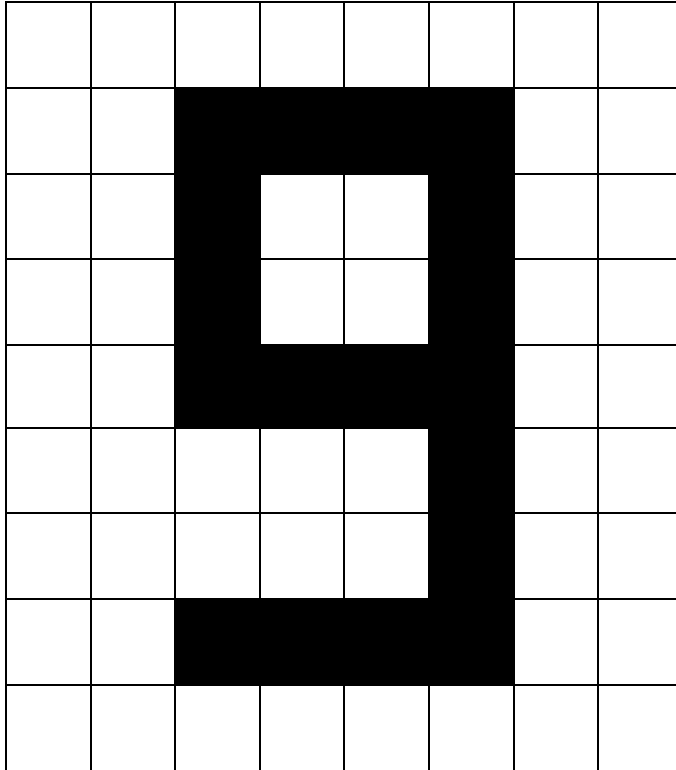
(c) Image

An image is a picture and it is stored on a computer. The image in a computer can know that means it has been changed into a collection of number that must be the computer can know well. In the images, a lot of pixel which is a tiny square of a color. A lot of these pixels can form an image. To store the picture in a computer, the computer can make simple record a number to describe the color of each square in a picture. To display an image on the screen, the computer tells the monitor to show a particular for each of the pixels. There are two methods in image that are called bitmap and vector. A bitmap is a procedure for storing images using pixels. That is called a bitmap because of a map of bits that are stored. Vector image are composed of lines, shapes, curves instead of the pixels. Vectors are especially used for making graphs and diagrams and it can't be used to store images in a computer.



(d) Graphics

Every computer also contains a lot of the pixels. Each of the pixel can make the image on the our computer's screen. It is saved in the computer as a bit pattern.



Matrix Representation

00000000

00111100

00100100

00100100

00111100

00000100

00000100

00111100

00000000

Linear Representation

00000000,00111100,00100100,00100100,00111100,00000100,
00000100,00111100,00000000

2. Convert the following binary number to ASCII character.

(a) 00100000_2

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

$$=0+0+32+0+0+0+0+0$$

$$=32_{10}$$

32_{10} converts into ASCII character is "space".

(b) 01011010_2

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

$$=0+64+0+16+8+0+2+0$$

$$=90_{10}$$

90_{10} converts into ASCII character is "Z" (capital Z).

(c) 00110000_2

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

$$=0+0+32+16+0+0+0+0$$

$$=48_{10}$$

90_{10} converts into ASCII character is "0" (zero).

3. Represent LITHAN as a binary number.

L = 76_{10}

76/2	38	0
38/2	19	0
19/2	9	1
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1

$76_{10} = 01001100_2$

$$I = 73_{10}$$

73/2	36	1
36/2	18	0
18/2	9	0
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1

$$73_{10} = 01001001_2$$

$$T = 84_{10}$$

84/2	42	0
42/2	21	0
21/2	10	1
10/2	5	0
5/2	2	1
2/2	1	0
1/2	0	1

$$84_{10} = 01010100_2$$

$$H = 72_{10}$$

72/2	36	0
36/2	18	0
18/2	9	0
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1

$$72_{10} = 01001000_2$$

$$A = 65_{10}$$

65/2	32	1
32/2	16	0
16/2	8	0
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

$$65_{10} = 01000001_2$$

$$N = 78_{10}$$

78/2	39	0
39/2	19	1
19/2	9	1
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1

$$78_{10} = 01001110_2$$

Therefore, LITHAN = 01001100₂ 01001001₂ 01010100₂ 01001000₂ 01000001₂ 01001110₂

4. Convert the following binary representation into text.

01001000 01100101 01101100 01101100 01101111

$$01001000 = 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$$

$$= 0 + 64 + 0 + 0 + 8 + 0 + 0 + 0$$

$$= 72_{10}$$

$$= H$$

$$\begin{aligned}01100101 &= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\&= 0 + 64 + 32 + 0 + 0 + 4 + 0 + 1 \\&= 101_{10} \\&= e\end{aligned}$$

$$\begin{aligned}01101100 &= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 \\&= 0 + 64 + 32 + 0 + 8 + 4 + 0 + 0 \\&= 108_{10} \\&= l\end{aligned}$$

$$\begin{aligned}01101100 &= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 \\&= 0 + 64 + 32 + 0 + 8 + 4 + 0 + 0 \\&= 108_{10} \\&= l\end{aligned}$$

$$\begin{aligned}01101111 &= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\&= 0 + 64 + 32 + 0 + 8 + 4 + 2 + 1 \\&= 111_{10} \\&= o\end{aligned}$$

Therefore, 01001000 01100101 01101100 01101100 01101111= Hello

P3: “Conversions of Numeric Data”

1. Converting 191_{10} to binary, octal and hexadecimal number system.

191_{10} to binary

	Quotient	Remainder
$191/2$	95	1
$95/2$	47	1
$47/2$	23	1
$23/2$	11	1
$11/2$	5	1
$5/2$	2	1
$2/2$	1	0
$1/2$	0	1

The answer is $191_{10} = 10111111_2$

191_{10} to octal

	Quotient	Remainder
$191/8$	23	7
$23/8$	2	7
$2/8$	0	2

The answer is $191_{10} = 277_8$

191_{10} to hexadecimal

	Quotient	Remainder
191/16	11	F
11/16	0	B

The answer is $191_{10} = BF_{16}$

2. Converting 1010110_2 to its decimal, octal and hexadecimal equivalent system.

1010110_2 to its decimal

$$\begin{aligned}
 1010110_2 &= 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\
 &= 64 + 0 + 16 + 0 + 4 + 2 + 0 \\
 &= 86_{10}
 \end{aligned}$$

The answer is $1010110_2 = 86_{10}$

Converting to octal

	Quotient	Remainder
86/8	10	6
10/8	1	2
1/8	0	1

The answer is $1010110_2 = 126_8$

Converting to hexadecimal

	Quotient	Remainder
86/16	5	6
5/16	0	5

The answer is $1010110_2 = 56_{16}$

3. Convert $BEEF_{16}$ to its decimal and binary equivalent system.

$BEEF_{16}$ to decimal

$$\begin{aligned}
 BEEF_{16} &= B \times 16^3 + E \times 16^2 + E \times 16^1 + F \times 16^0 \\
 &= 11 \times 16^3 + 14 \times 16^2 + 14 \times 16^1 + 15 \times 16^0 \\
 &= 45056 + 3584 + 224 + 15 \\
 &= 48879_{10}
 \end{aligned}$$

The answer is $BEEF_{16} = 48879_{10}$

Converting to binary

	Quotient	Remainder
48879/2	24439	1
24439/2	12219	1
12219/2	6109	1
6109/2	3054	1
3054/2	1527	0
1527/2	763	1
763/2	381	1
381/2	190	1
190/2	95	0
95/2	47	1
47/2	23	1
23/2	11	1
11/2	5	1
5/2	2	1
2/2	1	0
1/2	0	1

The answer is $BEEF_{16} = 1011111011101111_2$

4. Convert 721_8 to its decimal and binary equivalent system.

721_8 to decimal

$$721_8 = 7 \times 8^2 + 2 \times 8^1 + 1 \times 8^0$$

$$= 448 + 16 + 1$$

$$= 465_{10}$$

The answer is $721_8 = 465_{10}$

Converting to binary

	Quotient	Remainder
$465/2$	232	1
$232/2$	116	0
$116/2$	58	0
$58/2$	29	0
$29/2$	14	1
$14/2$	7	0
$7/2$	3	1
$3/2$	1	1
$1/2$	0	1

The answer is $721_8 = 111010001_2$

5. Add 35_{10} and 99_{10} using binary arithmetic and give the answer in denary form.

$$35_{10} = 100011_2$$

	Quotient	Remainder
35/2	17	1
17/2	8	1
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

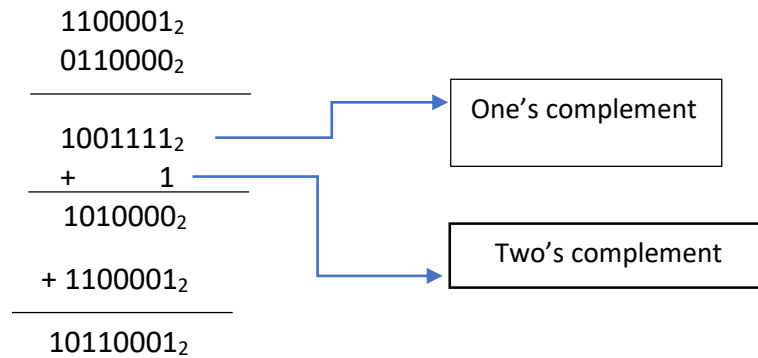
$$99_{10} = 1100011_2$$

	Quotient	Remainder
99/2	49	1
49/2	24	1
24/2	12	0
12/2	6	0
6/2	3	0
3/2	1	1
1/2	0	1

$$\begin{aligned}
 35_{10} + 99_{10} &= 100011_2 + 1100011_2 \\
 &= 10000110_2 \\
 &= 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\
 &= 128 + 0 + 0 + 0 + 0 + 4 + 2 + 0 \\
 &= 134_{10}
 \end{aligned}$$

The answer is $35_{10} + 99_{10} = 134_{10}$

6. Subtract 110000_2 from 1100001_2 using two's complement method and then convert to denary number system.



Converting to denary number system

$$110001_2 = 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$= 32 + 16 + 0 + 0 + 0 + 1$$

$$= 49_{10}$$

The answer is 49_{10}

7. Convert 2.625_{10} to our 8 bit binary floating point format.

2.625_{10}

	Quotient	Remainder
2/2	1	0
1/2	0	1

Multiply by 2 (Calculate the fraction part)

.625x2	1
.25x2	0
.5x2	1
0	

$$2.625 = 10.101_2$$

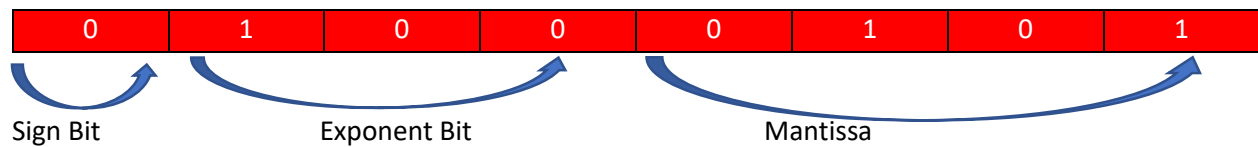
Floating point $10.101_2 = 1.0101 \times 2^1$

Exponent $= 1$

Exponent bit $= 3+1 = 4 = 100_2$

Mantissa $= 0101$

Sign bit $= 0$



8. Convert 39887.5625_{10} to IEEE 32-bit binary floating point format.

	Quotient	Remainder	Multiply by 2 (Calculate the fraction part)	
$39887/2$	19943	1	$.5625 \times 2$	1
$19943/2$	9971	1	$.8125 \times 2$	0
$9971/2$	4985	1	$.25 \times 2$	0
$4985/2$	2492	1	$.5 \times 2$	1
$2492/2$	1246	0	0	
$1246/2$	623	0		
$623/2$	311	1		
$311/2$	155	1		
$155/2$	77	1		
$77/2$	38	1		
$38/2$	19	0		
$19/2$	9	1		
$9/2$	4	1		
$4/2$	2	0		
$2/2$	1	0		
$1/2$	0	1		

$$39887.5625_{10} = 1001101111001111.1001_2$$

$$\text{Floating point } 1001101111001111.1001_2 = 1.0011011110011111001_2 \times 2^{15}$$

$$\text{Exponent} = 15$$

$$\text{Exponent bit } 127+15 = 142 = 10001110_2$$

$$\text{Mantissa} = 0011011110011111001$$

$$\text{Sign bit} = 0$$



9. Convert the 5.4375_{10} to fixed point binary number

	Quotient	Remainder
5/2	2	1
2/2	1	0
1/2	0	1

Multiply by 2 (Calculate the fraction part)	
.4375x2	0
.875x2	1
.75x2	1
.5x2	1
0	

The answer is $5.4375_{10} = 101.0111_2$

P4. Carry Out Boolean logic Operations.

Boolean logic Operations

Logic Operation is a phrase which connects two or extra terms of data. It is most usually used to check whether it has a certain value that are true and false. In programming, logic operations are necessary due to the fact they may be used to model the way that data flows through electric circuits, like circuits inner CPU. This type of operation is referred to as Boolean logic operations. A building block in a circuit due to the Boolean logic are called logic gates. A logic gate has as a minimum two inputs and one output. There are seven form of logic gates they are- AND, OR, XOR, NOT, NAND, and XNOR. among these seven gates, let's discover which gate should be used for building half adder circuit.

Type of gate- AND

Truth table for the AND gate

AND GATE		
INPUT		Output
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Here is the circuit diagram of the AND logic Gate.



Boolean Expression (A AND B) $A \cdot B$

This AND gate is included as the one part of the logic gates. In this logic gate, there are two values for the input one input might be false or 0 and the next input is false or 0. These means that if the two inputs are false, the output will be false or 0. And the another possible one is that if one put might be false or 0 and the next input might be true or 1. At this situation, the output will be recognized as a false or 0. If the two inputs are true Or 1, the output will be true or 1.

Type of gate- OR

Truth table for the OR gate

OR GATE		
INPUT		Output
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Here is the circuit diagram of the OR gate



Boolean Expression- $(A \text{ OR } B) A+B$

In the OR gate, there are two inputs and one output in it. It is known as a OR gate because the output will be high if the two inputs are high. This OR gate obey the logic operation of the input and output. If one of the input is true or 1 and the next one and the output will be 0. If both of the input values are true or 1 , the output will be true or 1.

Type of gate- EXOR

Truth table for the EXOR gate

EXOR GATE		
Input		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Here is the diagram of the EXOR gate



Boolean Expression- A XOR B ($A \oplus B$)

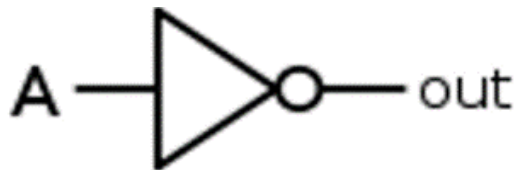
EXOR gate is also known as the Exclusive-OR gate. EXOR gate also have two inputs and one output like other gates. The value of the EXOR gate is if one of the input is true, the output will be true. If both of the inputs are true or false, the output will be false.

Type of gate- NOT

Truth table for the NOT gate

NOT GATE	
Input	Output
0	1
1	0

Here is the circuit diagram of the NOT gate



There are one input and one output in this NOT gate. This gate is also known as the inverter because it inverts both of the input and output values. AS an example if the input value is 1 and then it converts into the 0 as well as if the input value is 0 and it inverts into 1.

Type of gate- NAND

Truth table for the NAND gate

Input		Output
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

Here is the circuit diagram of the NAND gate



Boolean Expression – $\overline{A \cdot B}$

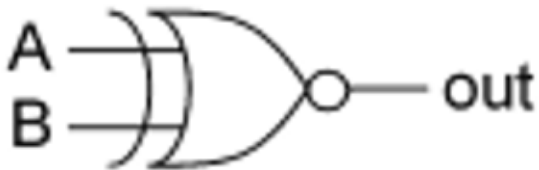
NAND gate is the combination of the AND gate and NOT gate. Inputs in the NAND gate can change if the both inputs value are false the output will be true. On the other hand, if one of the input are true or false and then the output value will be true. If the two input values are true, the output value will be false. In the diagram, a small circle on the output means the inversion.

Type of gate- EXNOR gate

Truth table for the EXNOR gate

Input		Output
A	B	A AND B
0	0	1
0	1	0
1	0	0
1	1	1

Here sis the circuit diagram for the EXNOR gate

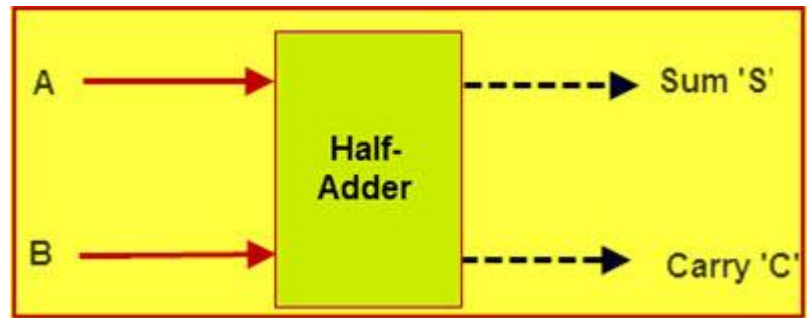
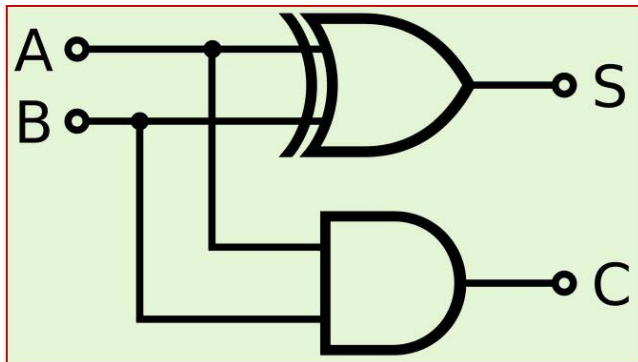


Boolean Expression – $\overline{A \oplus B}$

EXNOR gate is the combination of the XOR and NOT gate. There are two inputs and one output in this EXNOR gate. If the both of the input values are true, the output will be true as well as if the two inputs are false the output will be false. In other side, both of the inputs are true or false the output will be false.

Building a half- adder circuit

Here is the diagram of the half adder circuit



A half adder circuit is formed by the combining EXOR gate and the AND gate. It is a kind of the digital circuit that performs the additions of the two numbers. There are two inputs and two outputs in a half adder. Input are described as A and B and the outputs are described as Sum and Carry. EXOR is applied to both inputs to produce sum and AND gate is used to both inputs to produce carry. By using half adder circuit, we can design the simple addition with the help of the logic gates. The addition of the two number is not complex in half adder circuit.

$$0+0 = 0$$

$$0+1 = 1$$

$$1+0 = 1$$

$$1+1 = 10$$

The results are not complex 1+1 is 10, thus the sum result might be written as 2 bit.

$$0+0 = 00$$

$$0+1 = 01$$

$$1+0 = 01$$

$$1+1 = 10$$

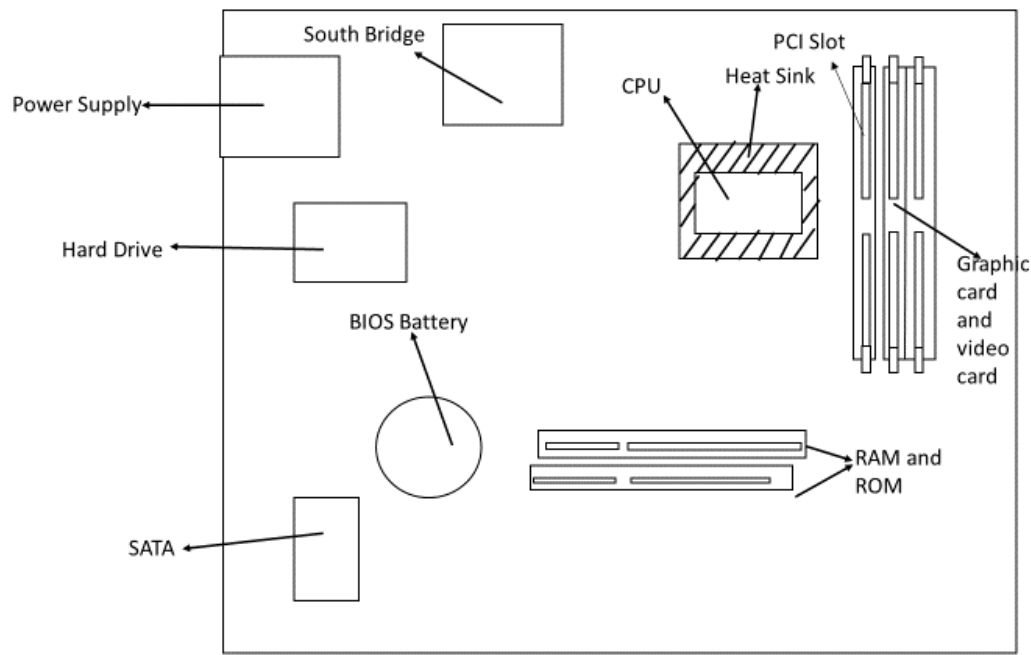
The output 1 of 10 becomes the carry and 0 will be sum value. The function is similar to the EXOR gate and the AND gate. For these two gates, the half adder circuit will be created. If the both input in the half adder circuit are false, both of the sum and carry value will become false. If one of input value is true and false, the value of the sum will be true and the carry value will be false. If both values are true, the output value will become different and the values of sum is false and the value of carry is true. If the two inputs are false, the outputs of the sum and the carry is false. If the one input is false and other input is true, the results of the Sum is true and the carry is false. If the one input is true and other input is false, the output results of the Sum is true and the carry is false. If the both of the input is false, the output result of the sum is false and carry is true.

Here is the truth table for the half adder circuit.

Input		Output	
A	B	SUM (S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

P5. “Key Computer System Components”

Here is the illustration of the computer system components.



In the above, we will see the motherboard and its components of the computer. Below is the description for each of the components.

1. Motherboard

Motherboard is the most complicated and important parts of the computer. A motherboard is situated at the back of the computer and that perform as a mainboard of the computer for the components. A computer has a various kind of components each of which has their own functions. The role of the motherboard is to allow all the components to communicate with each other. As the fact that all the other components are connected on the motherboard, it is secure to say that the motherboard is the essential part of a computer and the component that brings it all together.

2. Memory

Memory is also important not only for the human but also for the computer. For the human, it means the place or the precious memories we have to store in it. As a computer, memory is used to store the important data and information. So, we need a lot of memory for the computer to store. There are three types in the memory to store a lot of data. Three types of the memory are primary memory (used for storing the data and the instruction to process), secondary memory (used for storing permanent data), cache memory (used to speed up the CPU). In the primary memory, there are two types of memory also involved. These two types of memory are RAM and ROM.

3. Peripherals

Peripherals are the device that support the input and output devices to connect with the computer. There are two distinct types of peripherals they are input and output devices. Output devices can be a printer, monitor etc. Input devices can also be Mouse, Keyboard, Scanner etc. Some of the peripherals like touchscreens may integrate various devices into a single hardware component that can be used both as an input and output device.

4. Input devices

In the computer, input device may be any hardware device that sends data to the computer and that allows us to interact with it. The most frequently used for the input devices on a computer are keyboard and mouse. On the other hand, there are still a lot of devices that can also be used to insert the data into the computer.

5. Output devices

Output device in a computer can send the data from a computer to the other device. The most commonly used output devices are monitor and speakers and there are still other output devices. A computer without output devices it would not be able to get information out of the computer systems or use computer systems to control devices. All of the output devices are created to communicate with humans.

6. PCI Slot

The PCI means Peripheral Component Interconnect which is a computer bus. It assists as a connection between our computer's motherboard and any kind of connected hardware, transferring the data and power between the computer and the device. It was presented by the year of 1992. The PCI bus came in both 32 bit and 64 bits and both of these are used to attach hardware to a computer. As I observed that, found out the one recommendation that is getting more modern card if our computer's motherboard does not have a PCI slot.

7. BIOS

BIOS mean Basic Input /Output System is also a kind of chip that can be found on motherboard and it is used to boot up the computer devices. AS first, when we open our computer, the BIOS first initiate to check a hardware, termed as Power-On Self Post (POST) to determine if the attachments are working well or not. And then, it starts to load the operating system into our computer's RAM. On the other hand, it also manages the data flow between computer's operating system and connected devices like keyboard, mouse, hard disk etc. In addition, we all know that BIOS stores date, time and the system

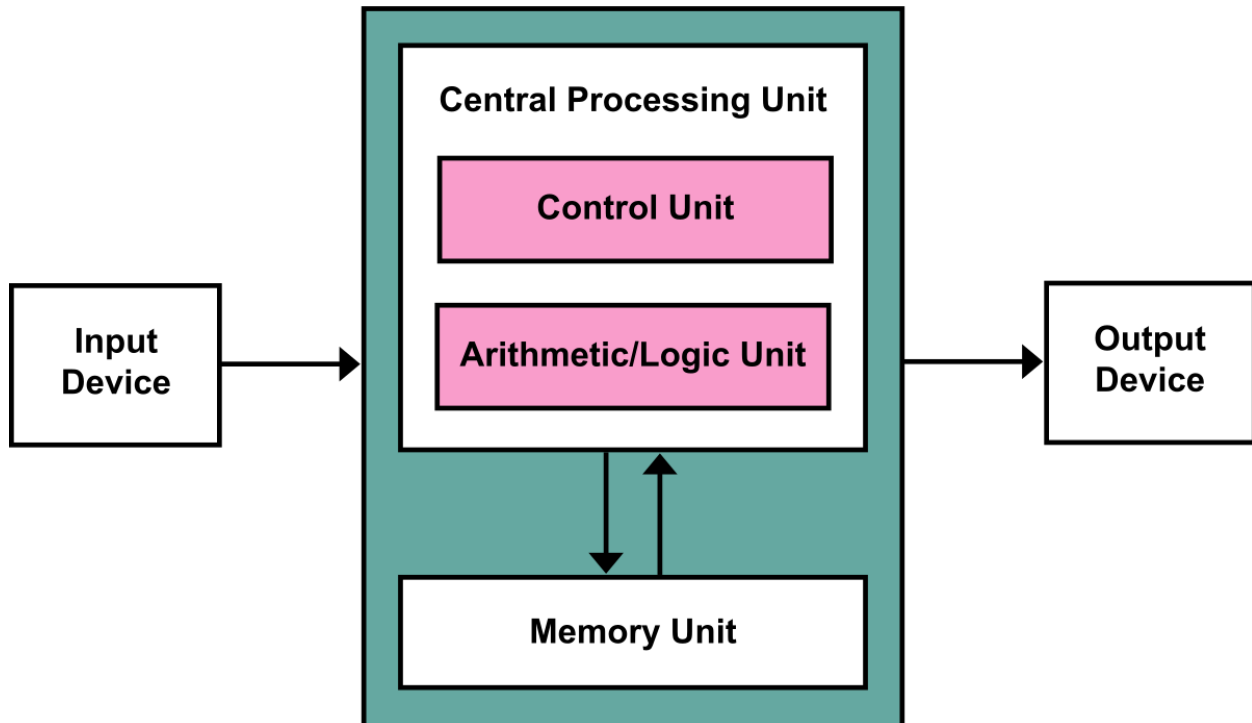
8. South Bridge

South Bridge is composed with two chips and it is situated on a computer of the motherboard. The south bridge naturally implements the lower capabilities of the motherboard in a south bridge chipset computer architecture. The south bridge can be classified from the north bridge by not being directly connected to the processor. The south bridge can manage all the input output functions like Audio, USB, BIOS etc.

9. SATA

SATA means Serial Advanced Technology Attachment and it is a kind of component that is concerned with the motherboard and situated in the motherboard. It is generally referred to the type of cable and used for transferring data between a computer's circuit board and storage devices. It was already designed to replace long-standing PATA (Parallel Advanced Technology Attachment) interface. SATA cable has a length of one meter and it have seven pins to connect with the motherboard.

Von Neumann Architecture



In the above photo, we can see that how CPU works and it is complex to execute the processes of a computer. In general, CPU has three components that are connecting input and output devices. These three components are-

- CU (Control Unit)
- Memory Unit
- ALU (Arithmetic /Logic Unit)

CPU (Central Processing Unit)

CPU that is also known as that essential part of the computer system and it controls the function of the other data and process. It is sometime called as a processor. It is a single integrated circuit that includes all the electronics needed to execute a program. It has a main work that is to control the data input the instructions that are to be performed on various kinds of the computers. In other way, it can create the communication of the peripherals in CPU with the buses which are data bus, control and address bus. CPU read the instructions from the memory (RAM) and then perform the instructions to become a computer program. CPU is situated in the mother board and if CPU isn't exist in our computer, all of the functions in the computer cannot work as well.

Control Unit(CU)

A control unit that leads operation within a processor of a computer. Control unit works by receiving the information of input and then converts into control signals. Then, these signals are sent to the central processor. After that the processor of the computer tells to the connected hardware what kind of operations to be carried out. The functions that a control unit performs are reliant on the CPU. And then, here is the functions of the control unit.

- At first, it collects the data operation from the users and then emit the signals to the various kinds of components.
- It controls the commands to transfer the data from input devices to an arithmetic logic unit.
- It also carries out all the results from the input devices to the memory to an arithmetic logic unit.
- And then, it also stored a program in the memory.
- In addition, it fetches the required instruction from the primary storage and decode each instruction and therefore execute them in a sequence.

Arithmetic-logical Unit (ALU)

AN arithmetic logic unit is that the part of the CPU that handles all the calculations the CPU might need. The Control unit supplies the data from the data that are required by the ALU from the memory or from input devices and directs the ALU to perform a unique operation based on the instructions that are fetched from the memory.

- ALU uses operands and code that tells it which operations to perform for input data.
- After the information is processed by the Alu, it is sent to the computer's memory.

Registers

Registers are used to store and transfer the data and instructions that are being used by the CPU. These registers are the best way to manipulate the data. In the simple processor, it consists of a single memory location, usually known as the accumulator. There are various types of the memories for the various reason. Some of the registers are Accumulator, Data Register, Memory Address register (MAR), Program Counter (PC), Memory Data Register (MDR), Instruction Register (IR), Memory Buffer Register (MBR).

Accumulator – It stores all the results and calculations that are made by the ALU.

Memory Address Register- It stores the memory locations of the instructions that are require to be fetched from memory or stored into memory.

Program Counter- It is a CPU register in the computer processor which the address of the next instruction executed from the memory.

Memory Data Register – The instructions are stored and fetched from memory and the data that need to transfer and store in a memory.

Instruction Register – It is an instruction that is fetched from the main memory. The control unit takes instruction from the register, decodes and executes it by sending signals to the suitable component of the computer to carry out the task.

Memory Buffer Register – This register holds the data or instruction read and written in the memory. The instruction located in this register are transferred to the IR, while the data are transferred to the accumulator or IR. In the other, we can say that this register is used to store the instruction of the data that are coming from the memory.

Memory Unit

Memory unit is a group of the storage together with associated circuits that are needed to transfer information in and out of the storage. The memory unit is a part of the computer and that handles all the data and instructions for processing. And it is closely connected with the CPU. Memory unit is composed to control the data and inserted instructions from the peripherals of the input.

Interfaces- It is a kind of network and it connects between the Output devices and CPU. In general, there are separated into five parts in the interfaces. These interfaces are Command line, Graphical user interface, Menu Driven (MDI), form based (FBI) and Natural language (NLI).

Clock - The clock is generally meaning ta microchip that controls the timing and speed of all the computer functions. It is also known as the crystal that vibrates an accurate frequency when the electricity is applied. The speed is measured in a clock speed like 1 MHZ is one million cycles, vibrations or a second. AS a 2GHZ is two billion cycles, vibrations or a second.

Buses- The buses are used to connect between the devices that is attached with the computer. There are three kind of buses they are – address bus, data bus, control bus. These buses enable a processor of a computer to communicate with the memory card or video card. The wire in the bus can carry a single bit of the information which means that if a bus has a lot of wires then it can address the more information.

Address Bus- This address bus carries the data information in the memory and the processor.

Data Bus – This data bus permits all of the data in a computer to store among the memory, input and output devices and the processor.

Control Bus- This bus carries and holds the commands from the CPU in order to control and organize all the activities within the computer.

P6: “Different Types of Memory”

1. DMA (Direct Memory Access)

Direct Memory Access is one of a system that is supplied by using the computer buses that allow data to send without delay to the attached device like disk drive to the memory at the mother board of the laptop. In other way, it can send a huge amount of the data between an external device and the main memory without CPU overall performance. DMA transfers are done by using a control circuit that is a part of the input and output device interface. As I observed that this circuit is known as the DMA controller. This DMA controller can carry out the functions that would usually be completed by the processor when accessing to the main memory. For each transferred, it provides the memory address and all of the bus signals that control the data transfer. since it has to transfer a huge amount of the data, the DMA controller should increment the memory address for successive words and keep track of the number of transfers. even as DMA is started taking place, the program which are requested to transfer can't continue and the processor may be used to execute another program. After executing the program, the processor can go back to the program which are requested to transfer. Input and output operations are done by means of the operation system in computer to response a request from the program. DMA can transfer the information without intervention by means of the processor, it must be the under the control of the program this is done with the aid of the processor. When the transfer is completed, DMA informs the processor with the aid of sending interrupt request. One point in DMA is that a difference among the processor and DMA controller can also rise if they are seeking to use the bus on the same time to access the main memory.

2. ROM (Read Only Memory)

As a read only memory, it could only be read and cannot be written. It is a kind of the primary memory. The data in this type of memory can be retained even though when the laptop power is switched off due to the fact it is a kind of non-volatile memory. In rom, the data can be modified to a limited number of times. The data that is required to be saved inside ROM is written throughout the manufacturing phase. It can save a program which are essential for the booting process of the computer. Instructions in Rom are built into the electric circuit of the chip. Those instructions are referred to as firmware. It can keep numerous megabytes of the data, typically 4MB or 8MB according to chip. In the laptop, the instructions are read from small program within the rom which are known as BIOS which means Basic Input and Output system. There are numerous rom chips which are located on the motherboard. Those chips are essential for the BIOS, bootup, reading and writing to the peripheral devices. There are three types in the ROM they are- PROM (Programmable Read Only Memory), EPROM (Erasable Programmable Read Only Memory), EEPROM (Electrically Erasable Programmable Read Only Memory).

PROM (Programmable Read Only Memory)	EPROM (Erasable Programmable Read Only Memory)	EEPROM (Electrically Erasable Programmable Read Only Memory)
It is a memory chip and the data can be written only at once on it. There is one point is that unlike RAM, PROM can retain the data and it is come out as a blank memory from the factory.	This memory can retain the data back until when it is exposed to ultra violet. This ultraviolet can remove the data and reprogram the memory.	This is a type of special memory and the data in this EEPROM can be erased by exposing it with an electric charge.

3. RAM (Random Access Memory)

RAM is also a type of the computer memory. This type of memory is called the Random-access memory and the main memory of the computer. This could be determined in the computers and other devices like phones, tablets and many others. It is a form of volatile memory and which can keep the data instructions and the results of the program that is currently carried out by way of the processor and the temporary data which is frequently used. In other way, it is known as the temporary memory due to the fact the data or information that are stored in RAM can lost if the power turned off in a computer. A data in the ram can be accessed randomly but it is expensive than different type of memory. Alternatively, it could be defined as a large of sequential locations of the memory which has a completely unique address determining the location and these locations in main memory are addressed directly by way of the instructions of the CPU. RAM has two types of memory- Dynamic Random Access Memory, Static Random Access Memory.

Static Random Access Memory (SRAM)	Dynamic Random Access Memory (DRAM)
SRAM is essential in a system to run optimally because it is very fast when compare to the DRAM. SRAM is widely used in electronics, microprocessor and in computing. When a data in SRAM no need to refreshed dynamically and it is still volatile that means when the power is removed from a memory of the device the data is not held and will appear. SRAM is faster, large in size, expensive than DRAM. It is also used as a cache memory.	DRAM must be continually refreshed in order to maintain the data. This is used for the most system memory and it is cheap and small in size. As I observed that all of the DRAMS are made up of the memory cells which are composed of one capacitor and one transistor. DRAM needs to be refreshed continuously, slow when compared to the SRAM. It is used as RAM and it is smaller in size.

4. Cache

The cache is the fastest memory when compare to the other memory. It can provide the access time of the data to the microprocessor of a computer with the high-speed. It is very expensive and integrated in the processor where most of the important data and programs are kept. Cache is also the volatile memory in computer which is attached closely to the CPU. So, it is called the memory of the CPU. The capacity of the cache is too low when compare to the random-access memory and Hard disk of a computer. The cache memory is situated within the processor and memory. The cache memory also stores some of the data in a short period of time. In the cache memory, there are classified in three levels which are Level (L1), Level (2) and Level (3).

For the L1, it is the primary type of the cache memory and it is fastest and nearest to the CPU. The size of the L1 is very small and when comparing to the others. It is between 2KB to 64KB. It is a fixed register in the CPU. If the CPU require the instructions it searched in L1 cache.

L2 is the second of the cache memory. L2 size is larger than L1 that is between 256 KB to 512KB. After searching the instruction in L1, if the require instruction is not found in L1 and then searched in L2.

L3 is the largest in size and it has the lowest speed when compare to L1 and L2. It size is within 1Mb to 8MB.

P7. Explain how polling and interrupts are used to allow communication between processor and peripherals.

There are two methods the first one is polling and the second is interrupt those methods are used to communicate among the processor and peripherals. A computer gets input from some of distinctive sources like character keys on the keyboard, mouse and data from the scanner. The arrival of this kind of input isn't always necessarily predicted at any precise time and the computer has to have a way of detecting it. In the first method polling, the processor continuously polls or checks each device in turn as to whether it requires attention. The polling method is executed with the aid of a polling program that shares processing time with the currently running task. A device indicates it requires attention by setting a bit in its device status register.

Most of the time, devices will no longer require attention and when one does it will must wait until it is next interrogated by the polling method. this is an inefficient method and much of the processors time is wasted on needless polls. In different way, polling has to test the peripheral (input and output devices) like keyboard, mouse, USB, hard drive, printer, scanner etc. Some of the advantages for the polling method is that it permits the computer to respond fast to the modified status of the input and output device as it is doing nothing else however waiting and asking over and over that will be fine if the CPU no need to do something else.

An interrupt that permits CPU to detect a tool while a device desires its attention. The CPU has an interrupt request wire line that is checked by the CPU after the execution of every single instruction. An interrupt is that lets in the processor to be executing its essential program and only stop to input and output devices. In reality, this method would provide an external input that would inform the processor that it should complete whatever the instruction that is currently running being accomplished fetch a new usual procedure that will provide the requesting device. Once this providing is finished, the processor would resume exactly where it left off. In an interrupt, there is an interrupt handler to handle the interrupts. as an example, if more than one interrupts are sent to the CPU after which the interrupt handler will handle the interrupts which are ready to be processed.

And last, polling and interrupt permit the CPU to stop the task this is currently running and send the reply to the essential task. Polling and interrupt are differing from every other in many aspects. But, how much the difference among the polling and interrupt there is but the one most important point that distinguishes is that in the polling method, CPU maintains on checking input and output devices at ordinary interval whether it needs CPU. In an interrupt the input and output device break the CPU and tell the CPU while the interrupt needs CPU. These are all about the interrupts and polling that are used among processor and peripherals. I have mentioned the details in above paragraphs.

P8. RISC and CISC

RISC (Reduced Instruction Set Computer)

RISC is reduced instruction set computer and it is also known as the kind of the microprocessor architecture that utilizes a small and highly optimized set of instructions. It is designed to perform a small types of computer instructions. It has a fixed number of instructions and they can execute all the instructions very rapidly because all the instructions are small and very simple. The chips in RISC need a few transistors these transistors can make the chips as a low price to design and produce. In the RISC, all the instructions set contain simple and basic instructions from these instructions a more complex instructions can be produced. All of the instructions can be completed in one cycle, which permits the processor to handle the instructions at the equal amount of time.

CISC (Complex Instruction Set Computer)

CISC is planned to complete all the tasks by using a smaller number of assembly lines. It is also known as the complex instruction. CISC contains a large number of complex instructions. CISC require more RAM to execute the instructions and all of these instructions cannot be completed in one clock cycle. Data transfer in CISC is from memory to memory. The CISC architecture in computer are designed to reduce the memory cost. In the other way, some of the programs need more storage for this reason the cost of the memory increase and the all the large memory becomes more expensive.

The uses of RISC and CISC is already mentioned in above explanations.

Advantages for RISC and CISC architecture

1. RISC processor has a simple design so it is the fastest processor than CISC processor.
2. When RISC is comparing to the CISC, the execution of the instructions in RISC processor is higher than due to the use of the registers to hold all the instructions.
3. RISC processor performance is often two or four times faster than that of the CISC processor due to the simplified instruction set.
4. RISC architecture use less chip space to reduce the instruction set.
5. In the CISC architecture, there is no need to change the instruction to add new commands into the chip.

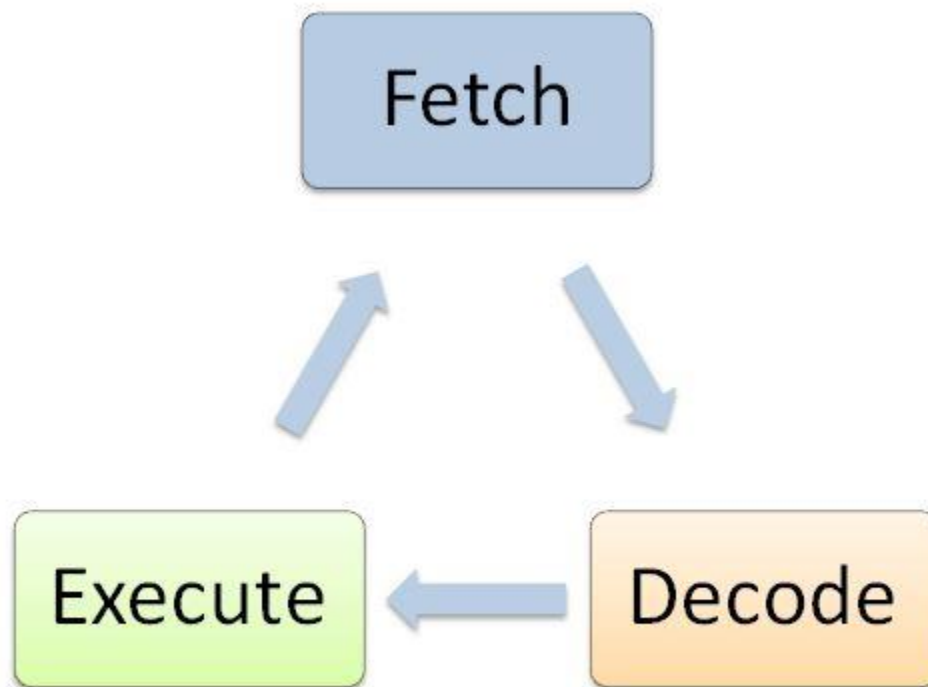
Disadvantages of RISC and CISC architecture

1. In the RISC, it requires the faster memory systems to execute the various kind of instructions.
2. Due to the performance of the machine slows down in CISC, the amount of the time taken for the various different instructions will be different.
3. CISC architecture require on-chip hardware to be continuously reprogrammed.
4. In RISC, the processor's performance based on the code that is being performed. The processor in the CISC expend much time for the first instruction result before it proceeds next instruction.
5. CISC chips require more transistors than RISC designs.

P9. “Fetch-Execute Cycle”

In a computer, there are a lot of instructions and functions that are inserted into the code by executing the binary numbering system. In order to perform these instructions, there is one of the ways and functions that can make this process works out is known as ‘Fetch-Decode-Execute’.

The summary steps for the fetch-decode-execute are as follows.



Fetch Cycle

The fetch cycle starts with the address that are stores in the program counter (PC). The address that are stores in the PC is some wrong address in the memory holding all the instructions to be executed. The CPU accomplished this step by fetching the instruction stored at the address from the memory and transferring the instruction to a special register IR to hold the instruction to be executed. The program counter is incremented to point to the next address from which the new instruction is to be fetched.

Decode Cycle

Another task for the process is to decode the instructions that are fetched in the fetch cycle. This process is known as the Decode. The CPU is already designed to make the instructions more understandable with a detailed set of commands and these are signified as the instruction set of the CPU. On the other hand, there are different types of commands in the CPU and then it decodes the commands and plan the several types of areas in the moment of the chip that are appropriate able for the next step.

Execute Cycle

This is the last process and al the instructions are successfully executed in this cycle and stored in the register called the Accumulator. After the execution is completed, and then CPU get ready to prepare again and again.

Memory Address Register- This register contains the value of the saved memory or the remaining values read from the memory. The instructions that are control in this register are moved to the IR while the data are transferred to the accumulator.

Memory Buffer Register- This register can handle the data or instruction and can be written in a memory. The instruction located in this register are transferred to the IR, while the data are transferred to the accumulator or IR.

Program Counter (PC) – This register is used for the execution for the next instruction. This register can support the next instruction of the address to be fetched.

Instruction Register – In this register, an instruction is fetched from the main memory and also stored in this register. The control unit denotes the instructions from this register, decode and executed.

Memory data register – It is a register of a computer control unit that includes the data to be stored in the storage of a computer. When the data is fetched from the memory, it is written in only one direction. When there is a write instruction in a register, the data that have been written is placed into another CPU register and which insert the data into the memory.

Accumulator – This register is made for using to keep up the data that are created by the system. After processing the results, all the results are stored in the AC register.

Accumulator register – this register is located in the ALU which is supported while the process of arithmetic and logical operations of ALU. This unit control controls the data that are fetched from main memory to perform arithmetic or logical operation in the accumulator. This register stimulates the initial data to be worked out the suddenly results and the last process. The final result is interchanged to main memory through the MBR.

Stack Pointer- This is a tiny register and stores the address of the final program in the stack which is recognized buffer that stores data from top to the down.

Index Register – Index Register is a register that can be signified by the instructions that use index addressing. It is usually controlled by one or more instructions with the skill to increment by an accurate amount.

M1 'Floating Point Numbers'

PC 122
Single Precision

Floating Point numbers are decided by using the IEEE standard normalized single precision. The floating point number in computer occupies the 32 bits of the numeric values. It is counted the number between 0 to 31 from the arrangement of left to right. It is used widely and the single precision is also known as in C, C++, C#, Java, Haskell, Pascal and Visual basic and MATLAB. The **MAR** 122 of the single precision floating point number is placed according to its wider over fixed point (in a fact that is referred to 'of the same band width').

CIR

Double Precision

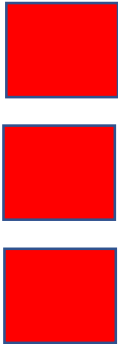
ACC Double Precision is an accurate variable precision numeric type. In other way, some of the values cannot be represented exactly and it is stored as the approximations. All of the double precision data types are 64bit IEEE standard. The 'double' is derived from the fact that a double precision number uses two times a bit a regular floating-point number. On the other way, it generally requires 32 bits for the double precision and then its double precision relevant will be added further.

Examples of the floating-point numbers for 32 bits

PC 123

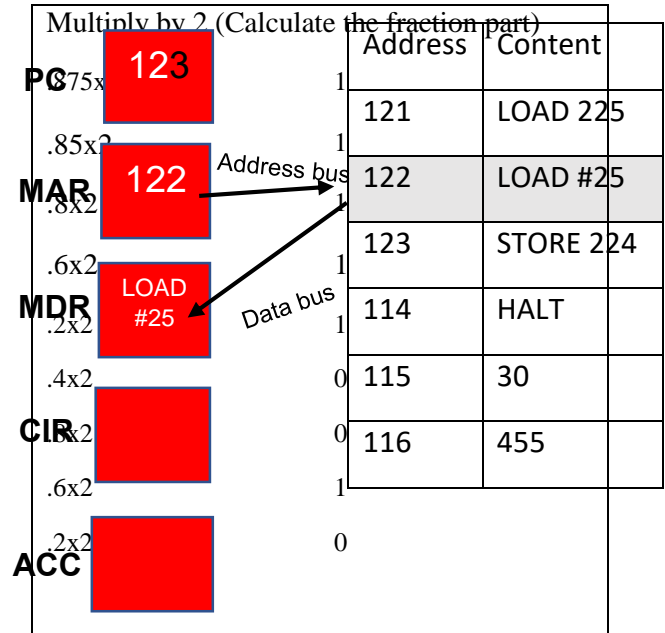
MAR 122

MAR << [PC]



Address	Content
121	LOAD 225
122	LOAD #25
123	STORE 224
124	HALT
125	30
126	455

MBR << [Memory] PC << [PC]+1

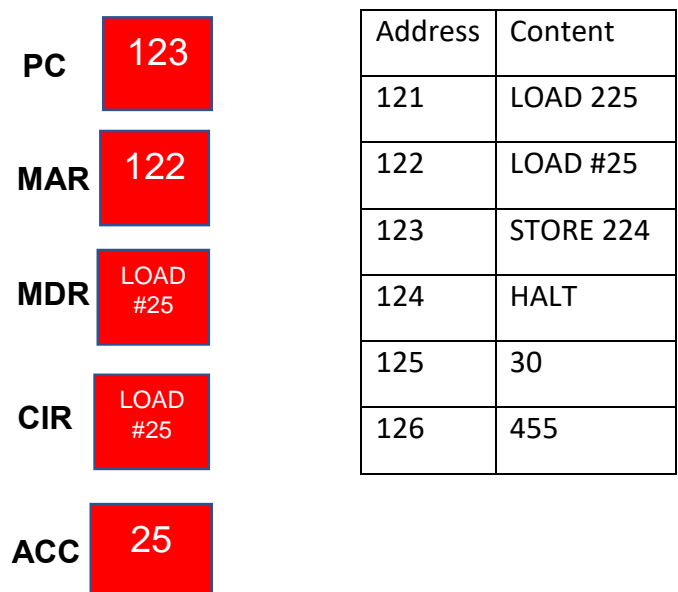


CIR << [MDR]



Address	Content
121	LOAD 225
122	LOAD #25
123	STORE 224
124	HALT
125	30
126	455

CIR Executed



Quotient

Remainder

4565/2	2282	1
2282/2	1141	0
1141/2	570	1
570/2	285	0
285/2	142	1
142/2	71	0
71/2	35	1
35/2	17	1
17/2	8	1
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

$4565.875 = 1000111010101.111110010_2$

Floating point: $1000111010101.111110010_2 = 1.000111010101111110010_2 \times 2^{12}$

Exponent: 12

Exponent bit: $127 + 12 = 139 = 10001011_2$

Mantissa: 000111010101111110010

Sign bit: 0

Sign bit	Exponent bit	Mantissa bit
0	10 001011	00011101010111111001000

Example for 16 bit

	Quotient	Remainder	Multiply by 2 (Calculate the fraction part)
55/2	27	1	.625x2 1

27/2	13	1	.25x2	0
13/2	6	1	.5x2	1
6/2	3	0	0	
3/2	1	1		
1/2	0	1		

55.625= 110111.101

Floating Point: $110111.101_2 = 1.10111101 \times 2^5$

Exponent: 5

Exponent Bit: $5+31=36=100100_2$

Mantissa: 10111101

Sign bit: 0

Sign bit	Exponent bit	Mantissa bit
0	100100	101111010

Example for the double floating-point number

	Quotient	Remainder	Multiply by 2 (Calculate the fraction part)	
4376/2	2188	0	.975x2	1
2188/2	1094	0	.95x2	1
1094/2	547	0	.9x2	1
574/2	287	0	.8x2	1
287/2	143	1	.6x2	1
143/2	71	1	.2x2	0
71/2	35	1	.4x2	0
35/2	17	1	.8x2	1
17/2	8	1	.6x2	1
			.2x2	0

<ol style="list-style-type: none"> 1. RAM has the accessible amount of speed to perform the tasks. 2. It is known as the fastest memory in a computer. 3. It is faster than the storage of the secondary. 4. It costs a few powers when compare to the hard disk drive so that can help the battery life increase. 	<ol style="list-style-type: none"> 1. When the computer power turned off, all of the information in the computer can be lost. 2. As I observed that, it is slower than the CPU cache memory. 3. There is a limited space to carry out the tasks.
--	---

ROM (Read Only Memory)

ROM stands for read only memory. The memory which can only read and but cannot write on it. This is a kind of non-volatile memory. The data and the contents of the information is kept permanently the moment of process of the manufacture. On the other way, it stores a lot of instructions that are require to start our computer. ROM are not only used in the computers but also used in the other electronic devices. If there is a few mistakes in a process of manufacture in ROM, it become useless. ROM is a non-volatile memory and the data in the ROM cannot be lost when the power is turned off.

Advantages of ROM	Disadvantages of ROM
<ol style="list-style-type: none"> 1. In the ROM, the data are written at once it will be forever. 2. The price of the ROM is cheaper than when compare to the RAM. 	<ol style="list-style-type: none"> 1. ROM cannot be support at all kinds of devices. 2. The data in the ROM cannot be change by the user's decision.

3. It can keep the data as permanent in the ROM.	3. It is the permanent memory and cannot be changed .
--	---

Cache Memory

Cache memory is the smallest and fastest than the other types of memory. The CPU can access it more quickly than the primary memory. It handles the data and instructions that are frequently used by the CPU and stored in a cache. It does not need to access the primary memory (RAM). It is already designed to speed up the data transfer between data and instructions. A copy of the data and the instructions that are recover from the RAM and when CPU uses for first time. The cache is the nearest to the CPU and it is always faster and store less than other type pf memory.

Advantages of cache memory	Disadvantages of cache memory
1. Cache memory stores the data within a short time.	1. It is too small in size and we cannot store a lot of data on it.

<ul style="list-style-type: none">2. It is faster when compare to the other types of memory.3. It costs less times to execute program when compared to the main memory.4. It can only be stored the data for a temporary.	<ul style="list-style-type: none">2. We cannot store the data permanently in this memory and the data are kept last longer as the power supply is supported.3. It costs a lot of money to use in reality.
---	--

M3. “Low-level Program”

Here is the code of the assembly

```
.model small
.stack 100h

.data

msg1 db ,0dh,0ah, "Enter 1st number with + or - sign : $"
msg2 db ,0dh,0ah, "Enter 2nd number with + or - sign : $"
msg3 db ,0dh,0ah, "The sum of two number is : $"
msg4 db ,0dh,0ah, "The product of two number is : $"
msg5 db ,0dh,0ah, " ~ Thank You ~ $ "
sign db ?, "$"
sign1 db ?, "$"
```

```
sign2 db ?, "$"
ans1 db ?, "$"
ans2 db ?, "$"
num1 db ?, "$"
num2 db ?, "$"

.code
main proc

mov ax,@data          ;initiaize ds
mov ds,ax

LOOP1:

mov dx,offset msg1     ;load and prompt 1st number
mov ah,09
int 21h

mov ah,1h              ;read sign of 1st number
int 21h

mov sign1,al

cmp al,2Dh
je N1

cmp al,2Bh
je N1

cmp al,2Bh
jne LOOP1

N1:

mov ah,1h              ;read magnitute of 1st number
int 21h

sub al,30h
mov num1,al

LOOP2:
```

```
mov dx,offset msg2      ;load and prompt 2nd number
mov ah,09
int 21h

mov ah,1h               ;read sign of 2nd number
int 21h

mov sign2,al

cmp al,2Dh
je N2

cmp al,2Bh
je N2

cmp al,2Bh
jne LOOP2

N2:

mov ah,1h               ;read magnitude of 2nd number
int 21h

sub al,30h
mov num2,al

mov dx,offset msg3      ;load and display sum result msg
mov ah,09
int 21h

mov bl,sign1
mov al,sign2

cmp bl,al               ;decision making on whether 2 numbers for sum have the
same sign
je S_Same               ;if 2 numbers for sum have the same sign, branch to
S_Same function

cmp bl,al
jne S_Diff              ;decision making on whether 2 numbers for sum have the
different sign
```

```
                                ;if 2 numbers for sum have the different sign, branch to
S_Diff function
S_Result:

mov dx,0000h
mov dl, offset sign      ;load and display sign of sum
mov ah,09
int 21h

mov dx,0000h
mov dl,offset ans1      ;load and display 1st digit of sum
mov ah,09
int 21h

mov dx,0000h
mov dl,offset ans2      ;load and display 2nd digit of sum
mov ah,09
int 21h

mov dx,offset msg4      ;load and display multiplication result msg
mov ah,09
int 21h

mov bl,sign1
mov al,sign2

cmp bl,al                ;decision making on whether 2 numbers for multiply have
the different sign
jne M_Diff               ;if 2 numbers for multiply have the different sign,
branch to M_Diff function

cmp bl,al                ;decision making on whether 2 numbers for multiply have
the same sign
je M_Same                ;if 2 numbers for multiply have the same sign, branch to
M_Same function

M_Result:

mov dx,0000h
mov dl, offset sign      ;display the sign of multiply result
mov ah,09
int 21h
```



```
mov dx,0000h
mov dl,offset ans1      ;display the 1st digit of multiply result
mov ah,09
int 21h

mov dx,0000h
mov dl,offset ans2      ;display the 2nd digit of multiply result
mov ah,09
int 21h

jmp Thank               ;jump to Thank You message

S_Same:                 ;Addition of same sign number function

mov al, num1
add al, num2
mov ah,00h
aaa

add ax,3030h
mov ans1,ah
mov ans2,al

mov dl,sign2
mov sign, dl

jmp S_Result           ;jump to display sum result function

S_Diff:                 ;Addition of different sign number function

mov al, num1
mov bl, num2

cmp al, bl
jl Sign_B

mov al, num1
sub al, num2
mov ah,00h
aaa
```

```
add ax,3030h
mov ans1,ah
mov ans2,al

jmp S_Result          ;jump to display sum result function

Sign_B:
mov al, num2
sub al, num1
mov ah,00h
aaa

add ax,3030h
mov ans1,ah
mov ans2,al

mov al, num1
cmp al, num2
jl Sign_A

Sign_A:
mov al,sign2
mov sign, al

jmp S_Result

M_Same:                ;Multiply of same sign number function

mov dh,00h
mov dl, num1
mov al, num2
mul dl

aam
add ax,3030h
mov ans1,ah
mov ans2,al

mov sign1, 2bh

jmp M_Result          ;jump to display multiply result function
```

```
M_Diff:                ;Multiply of different sign number function

mov dh,00h
mov dl, num1
mov al, num2
mul dl

aam
add ax,3030h
mov ans1,ah
mov ans2,al

mov al,2dh
mov sign,al

jmp M_Result          ;jump to display multiply result function

Thank:
mov dx,offset msg5    ;load and display Thank You message
mov ah,09
int 21h

main endp
end main
```

emu8086 - assembler and microprocessor emulator 4.08

file edit bookmarks assembler help

NEW OPEN SAVE ASSEMBLE RUN

```
0002 sub al,30h
0003 mov num2,al
0004
0005
0006
0007 mov dx,offset msg3 ;load and display sun result msg
0008 mov ah,09
0009 int 21h
0010
0011
0012 mov bl,sign1
0013 mov al,sign2
0014
0015 cmp bl,al
0016 je S_Same
0017
0018 cmp bl,al
0019 jne S_Diff
0020
0021 S_Result:
0022
0023 mov dx,0000h
0024 mov dl,offset sign ;load and display sign of sun
0025 mov ah,09
0026 int 21h
0027
0028 mov dx,0000h
0029 mov dl,offset ans1 ;load and display 1st digit of sun
0030 mov ah,09
0031 int 21h
0032
0033 mov dx,0000h
0034 mov dl,offset ans2 ;load and display 2nd digit of sun
0035 mov ah,09
0036 int 21h
0037
0038 mov dx,offset msg4 ;load and display multiplication result msg
0039 mov ah,09
0040 int 21h
0041
0042 mov bl,sign1
0043 mov al,sign2
0044
0045 cmp bl,al
0046 jne M_Diff
0047
0048 cmp bl,al
0049 je M_Same
0050
0051 M_Result:
0052
0053 mov dx,0000h
0054 mov dl,offset sign ;display the sign of multiply result
0055 mov ah,09
0056 int 21h
0057
```

These are the decision making

emu8086 - assembler and microprocessor emulator 4.08

file edit bookmarks assembler help

NEW OPEN SAVE ASSEMBLE RUN

```
147 mov dx,0000h
148 mov dl,offset ans2 ;display the 2nd digit of multiply result
149 mov ah,09
150 int 21h
151
152 jmp Thank
153
154
155 S_Same:
156 ;Addition of same sign number function
157
158 mov al,num1
159 add al,num2
160 mov ah,00h
161 aaa
162 add ax,3030h
163 mov ans1,ah
164 mov ans2,al
165
166 mov dl,sign2
167 mov sign,d1
168
169 jmp S_Result
170
171
172 S_Diff:
173 ;Addition of different sign number function
174
175 mov al,num1
176 mov bl,num2
177 cmp al,bl
178 jl Sign_B
179
180 mov al,num1
181 sub al,num2
182 mov ah,00h
183 aaa
184
185 add ax,3030h
186 mov ans1,ah
187 mov ans2,al
188
189 jmp S_Result
190
191
192 Sign_B:
193 mov al,num2
194 sub al,num1
195 mov ah,00h
196 aaa
197
198 add ax,3030h
199 mov ans1,ah
200 mov ans2,al
201
202 mov al,num1
203 cmp al,num2
204 jl Sign_A
```

Branching

emu8086 - assembler and microprocessor emulator 4.08

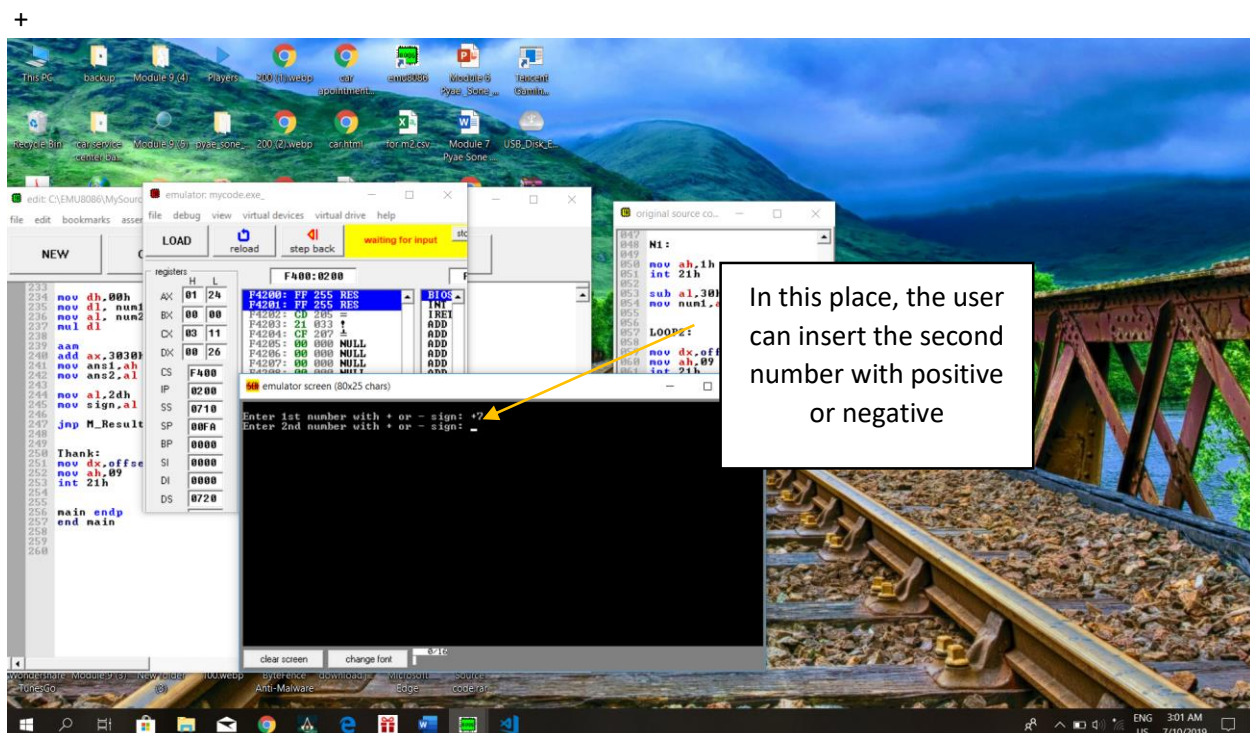
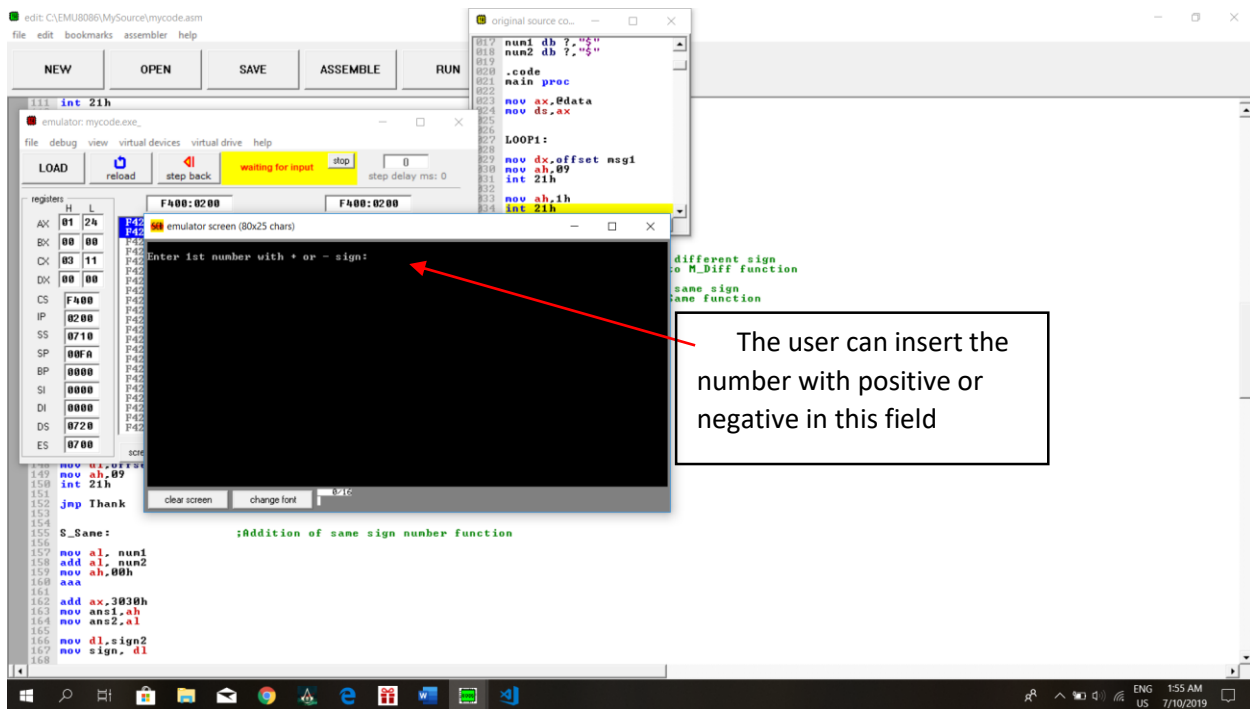
file edit bookmarks assembler help

NEW OPEN SAVE ASSEMBLE RUN

```
111 int 21h
112
113
114 mov dx,0000h
115 mov dl,offset ans2 ;load and display 2nd digit of sum
116 mov ah,09
117 int 21h
118
119
120 mov dx,offset msg4 ;load and display multiplication result msg
121 mov ah,09
122 int 21h
123
124 mov bl,sign1
125 mov al,sign2
126
127 cmp bl,al
128 jne M_Diff ;decision making on whether 2 numbers for multiply have the different sign
129 ;if 2 numbers for multiply have the different sign, branch to M_Diff function
130 cmp bl,al
131 je M_Same ;decision making on whether 2 numbers for multiply have the same sign
132 ;if 2 numbers for multiply have the same sign, branch to M_Same function
133
134 M_Result:
135
136 mov dx,0000h
137 mov dl,offset sign ;display the sign of multiply result
138 mov ah,09
139 int 21h
140
141 mov dx,0000h
142 mov dl,offset ans1 ;display the 1st digit of multiply result
143 mov ah,09
144 int 21h
145
146
147 mov dx,0000h
148 mov dl,offset ans2 ;display the 2nd digit of multiply result
149 mov ah,09
150 int 21h
151
152 jmp Thank ;jump to Thank You message
153
154
155 S_Same: ;Addition of same sign number function
156
157 mov al,num1
158 add al,num2
159 mov ah,00h
160 aas
161
162 add ax,3030h
163 mov ans1,ah
164 mov ans2,al
165
166 mov dl,sign2
167 mov sign,dl
168
```

Assembly code comment

Windows taskbar: 1:49 AM, 7/10/2019, ENG US



In this photo, after inserting first and second number the addition and the production of the two number results are come out as shown in below.

