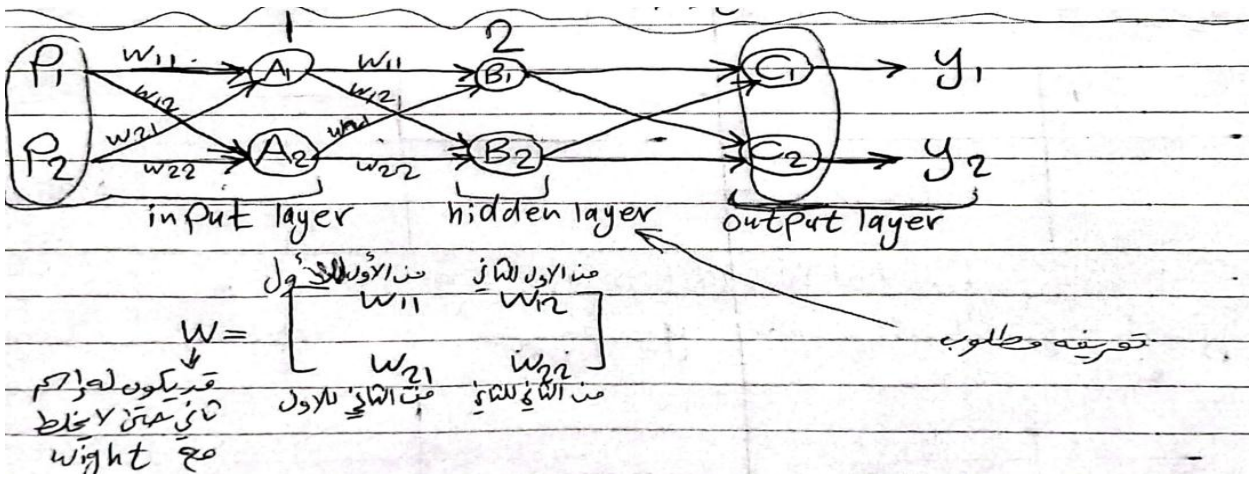
Q/ Draw a neural network with one hidden layer each layer Contai
2 neurons ?

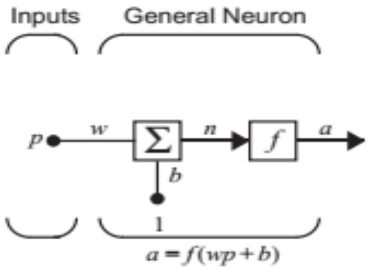


H.W

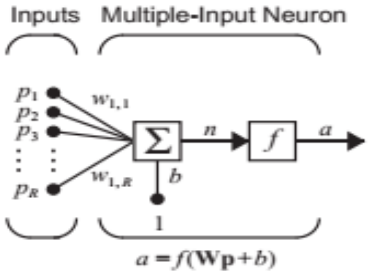
Q/ Draw a neural network with 2 hidden layrs each layer contains 3 neurons ?

Summary of Results

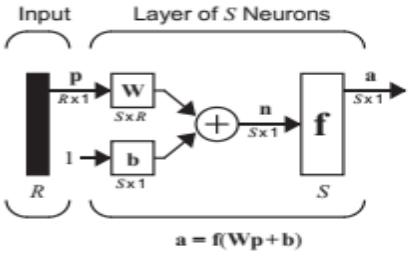
Single-Input Neuron



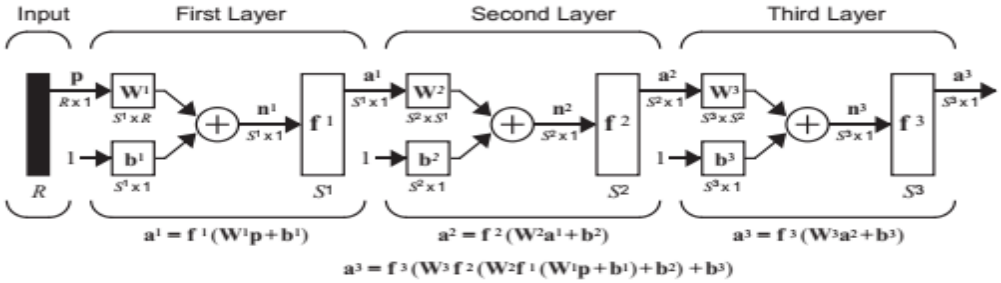
Multiple-Input Neuron



Layer of Neurons



Three Layers of Neurons

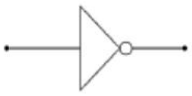
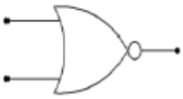


البوابة المنطقية (Logic Gates):

عبارة عن عنصر إلكتروني رقمي يمثل وحدة البناء الأساسية في الأنظمة الرقمية و يقوم بتنفيذ تابع منطقي معين.

تقسم البوابات المنطقية إلى: البوابات المنطقية الأساسية وهي تضم بوابات NOT, AND, OR وإلى بوابات المستوى الثاني وهي بوابات NAND, NOR, XOR, XNOR .

الجدول الكامل للبوابات المنطقية: يظهر الجدول رمز كل بوابة، مع التابع المنطقي الخاص بها، وجدول الحقيقة الذي يصف عملها

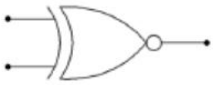
اسم البوابة	الرمز المنطقي	التابع المنطقي	جدول الحقيقة
بوابة النفي (العاكس) NOT		$F=\overline{x}$	x y 0 1 1 0
بوابة الجمع المنطقي OR		$F=x+y$	x y x+y 0 0 0 0 1 1 1 0 1 1 1 1
بوابة الضرب المنطقي		$F=x.y$	x y x.y 0 0 0 0 1 0 1 0 0 1 1 1
بوابة نفي الجمع NOR		$F=\overline{(x+y)}$	x y $\overline{(x+y)}$ 0 0 1 0 1 0 1 0 0 1 1 0
بوابة نفي الضرب NAND		$F=\overline{(x.y)}$	x y $\overline{(x.y)}$ 0 0 1 0 1 1 1 0 1 1 1 0

بوابة الجمع الحصري
XOR

$F = x \oplus y$
$F = (x.\overline{y}) + (\overline{x}.y)$

x	y	x⊕y
0	0	0
0	1	1
1	0	1
1	1	0

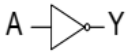
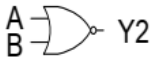
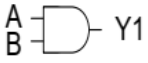
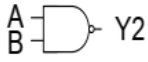
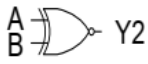
بوابة نفي الجمع الحصري
XNOR



$F = \overline{(x \oplus y)}$
$F = (x.y) + (\overline{x}.\overline{y})$

x	y	$\overline{(x \oplus y)}$
0	0	1
0	1	0
1	0	0
1	1	1

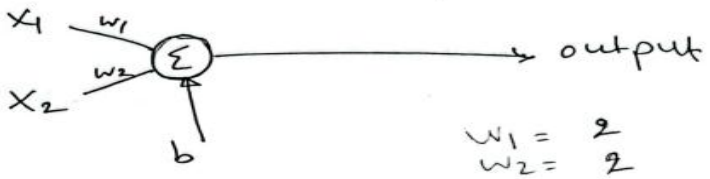
The seven gates that are the fundamental logic elements in digital system are illustrated in Fig. (1.1)

Logic Function	Logic Gate Symbol	Truth Table	Boolean Expression
NOT		Input (A) / Output (Y): 0 → 1; 1 → 0	$Y=\overline{A}$
OR		Input A, B / Output Y1, Y2: 0 0 → 0 1; 0 1 → 1 0; 1 0 → 1 0; 1 1 → 1 0	Y1= A+B
NOR			$Y2= \overline{A+B}$
AND		Input A, B / Output Y1, Y2: 0 0 → 0 1; 0 1 → 0 1; 1 0 → 0 1; 1 1 → 1 0	Y1= A.B
NAND			$Y2= \overline{A.B}$
EX-OR		Input A, B / Output Y1, Y2: 0 0 → 0 1; 0 1 → 1 0; 1 0 → 1 0; 1 1 → 0 1	Y1= A⊕ B
EX-NOR			$Y1= \overline{A \oplus B}$

EX.

$$O/P = \begin{cases} 1 & xw+b \geq 0 \\ 0 & xw+b < 0 \end{cases}$$

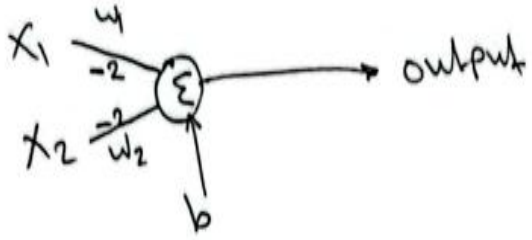
AND gate :



$w_1 = 2$
$w_2 = 2$
$b = -3$

x_1	x_2	computational	o/p
0	0	$0\times2+0\times2-3=-3$	0
0	1	$0\times2+1\times2-3=-1$	0
1	0	$1\times2+0\times2-3=-1$	0
1	1	$1\times2+1\times2-3=1$	1

H.W



$$O/P = \begin{cases} 1 & xw+b \geq 0 \\ 0 & xw+b < 0 \end{cases}$$

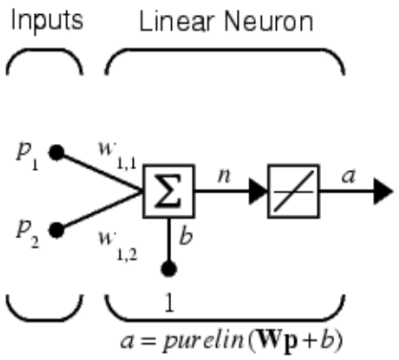
NAND gate

x_1	x_2	computation	o/p

x_1	x_2	b
-2	-2	3

EXP (1)

Write a neural network for the following



$$\mathbf{W} = \begin{bmatrix} 1 & 2 \end{bmatrix} \text{ and } b = \begin{bmatrix} 0 \end{bmatrix}$$

\

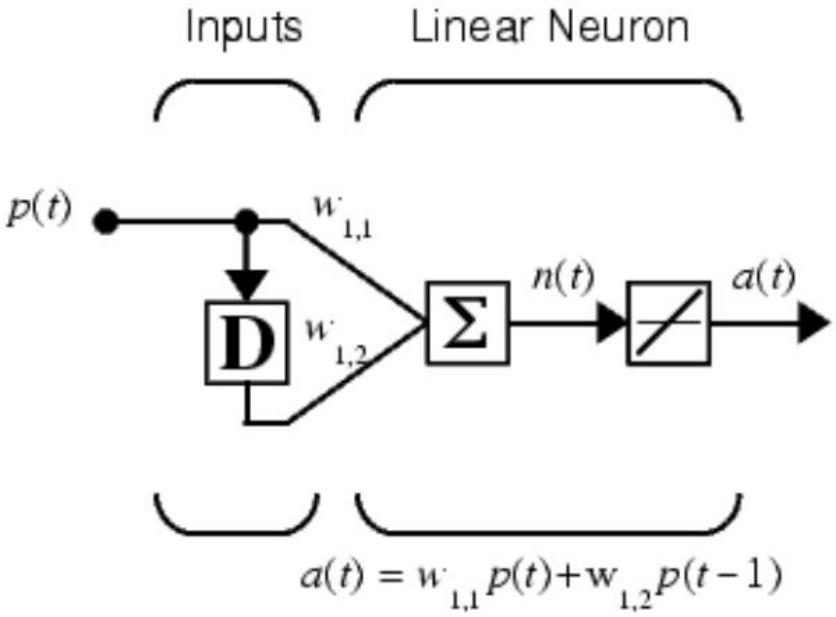
$$\mathbf{p}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \mathbf{p}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \mathbf{p}_3 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad \mathbf{p}_4 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Sol.

```
net = linearlayer;
net.inputs{1}.size = 2;
net.layers{1}.dimensions = 1;

 net.IW{1,1} = [1 2];
 net.b{1} = 0;

P = [1 2 2 3; 2 1 3 1];

A = net(P)
A =
     5     4     8     5
```

EXP (2)

Write a neural network for the following



$$\mathbf{W} = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

$$\mathbf{p}_1 = \begin{bmatrix} 1 \end{bmatrix}, \quad \mathbf{p}_2 = \begin{bmatrix} 2 \end{bmatrix}, \quad \mathbf{p}_3 = \begin{bmatrix} 3 \end{bmatrix}, \quad \mathbf{p}_4 = \begin{bmatrix} 4 \end{bmatrix}$$

Sol

```
net = linearlayer([0 1]);
net.inputs{1}.size = 1;
net.layers{1}.dimensions = 1;
net.biasConnect = 0;
```

```
net.IW{1,1} = [1 2];
```

```
P = {1 2 3 4};
```

```
A = net(P)
A =
    [1]    [4]    [7]    [10]
```

3.2 McCulloch-Pitts Neuron Model

The first formal definition of a synthetic neuron model based on the highly simplified considerations of the biological model was formulated by Warren McCulloch and Walter Pitts in 1943. The McCulloch-Pitts model of a neuron is characterized by its formalism, elegant, and precise mathematical definition.

McCulloch-Pitts neuron allows binary 0 or 1 states only, i.e. it is binary activated. These neurons are connected by direct weighted path. The connected path can be excitatory or inhibitory. Excitatory connections have positive weights and inhibitory connections have negative weights. There will be same weights for the excitatory connection entering into a particular neuron. The neuron is associated with the threshold value. The neuron fires if the net input to the neuron is greater than the threshold. The threshold is set so that the inhibition is absolute, because, non-zero inhibitory input will prevent the neuron from firing. It takes only one time step for a signal to pass over one connection link.

3.2.1 Architecture

The architecture of the McCulloch Pitts neuron is shown in Fig. 3.1.

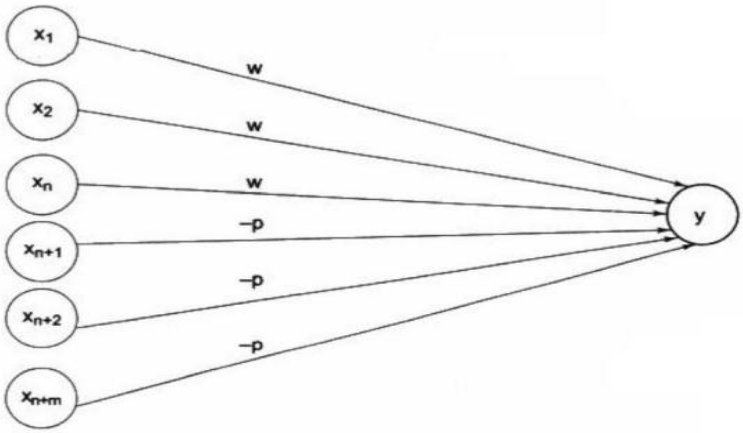


Fig. 3.1 *Architecture of a McCulloch-Pitts Neuron*

'Y' is the McCulloch-Pitts neuron, it can receive signal from any number of other neurons. The connection weights from $x_1, \ldots x_n$ are excitatory, denoted by 'w' and the connection weights from

$x_{n+1} \cdots x_{n+m}$ are inhibitory denoted by '– p'. The McCulloch-Pitts neuron Y has the activation function,

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{-in} \geq \theta \\ 0 & \text{if } y_{-in} < \theta \end{cases}$$

where θ is the threshold and y_{-in} is the total net input signal received by neuron Y.

The threshold θ should satisfy the relation.

$$\theta > nw - p$$

This is the condition for absolute inhibition.

The McCulloch-Pitts neuron will fire if it receives k or more excitatory inputs and no inhibitory inputs, where

$$k_w \geq \theta > (k - 1)\, w.$$

Example 3.1 Generate the output of logic AND function by McCulloch-Pitts neuron model.

Solution The AND function returns a true value only if both the inputs are true, else it returns a false value. '1' represents true value and '0' represents false value.

The truth table for AND function is,

x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

A McCulloch-Pitts neuron to implement AND function is shown in Fig. 3.2. The threshold on unit Y is 2.

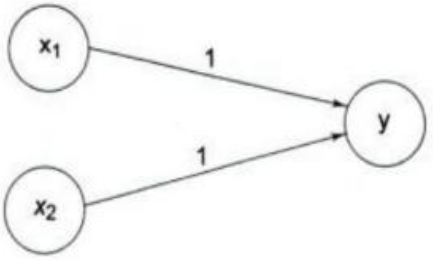


Fig. 3.2 *McCulloch-Pitts Neuron to Perform Logical AND Function*

The output Y is,

$$Y = f(y_{in})$$

The net input is given by

$$y_{in} = \sum_i \text{weights} * \text{input}$$

$$y_{in} = 1 * x_1 + 1 * x_2$$

$$y_{in} = x_1 + x_2$$

From this the activations of output neuron can be formed.

$$Y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{-in} \geq 2 \\ 0 & \text{if } y_{-in} < 2 \end{cases}$$

Now present the inputs

(i) $x_1 = x_2 = 1, \quad y_{in} = x_1 + x_2 = 1 + 1 = 2$

$y = f(y_{in}) = 1$ since $y_{in} = 2$.

(ii) $x_1 = 1, x_2 = 0, y_{in} = x_1 + x_2 = 0 + 1 = 1$

$y = f(y_{in}) = 0$ since $y_{in} = 1 < 2$.

This is same when $x_1 = 0$ and $x_2 = 1$.

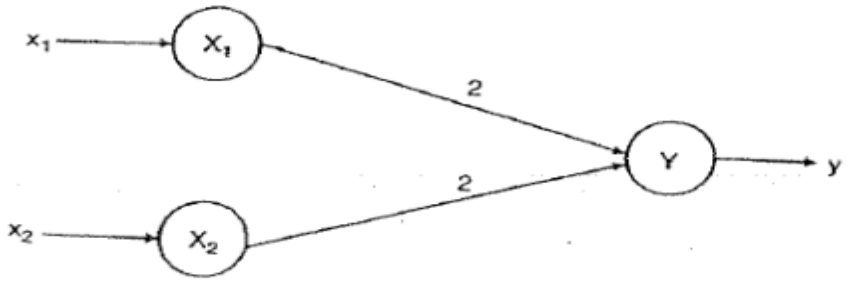
(iii) $x_1 = 0, \ x_2 = 0, \ y_{in} = x_1 + x_2 = 0 + 0 = 0$.

Hence, $y = f(y_{in}) = 0$ since $y_{in} = 0 < 2$.

H.W

Q) Solve OR gate by using McCulloch-Pitts method.

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	1



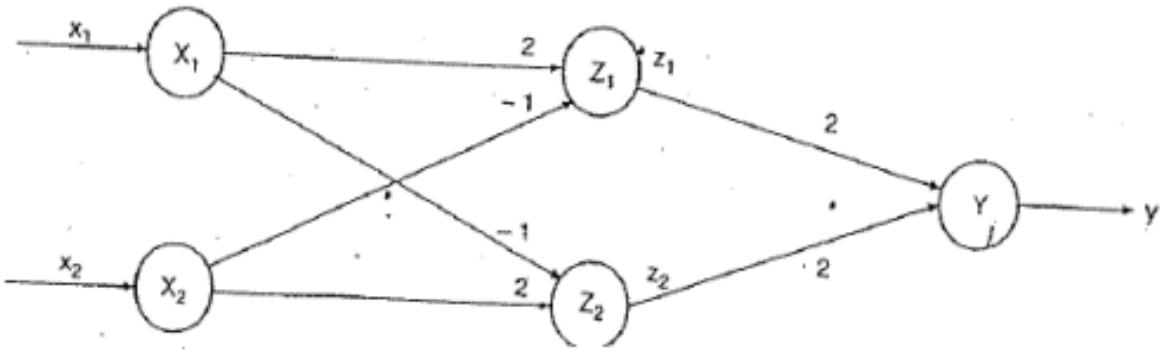
The output of the neuron Y is written as:

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 2 \\ 0 & \text{if } y_{in} < 2 \end{cases}$$

Q) Solve XOR gate by using McCulloch-Pitts method.

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Solution:



The output of the neuron z_i $(i = 1, 2)$ is:

$$z_i = f(z_{in}) = \begin{cases} 1 & \text{if } z_{in} \geq 2 \\ 0 & \text{if } z_{in} < 2 \end{cases}$$

The output of the neuron Y is:

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 2 \\ 0 & \text{if } y_{in} < 2 \end{cases}$$

where $zin_1 = 2x_1 - x_2$,

$zin_2 = 2x_2 - x_1$

and $y_{in} = 2z_1 + 2z_2$.

If $x_1 = 1$ and $x_2 = 0$

then

$$z_{in1} = 2x_1 - x_2 = 2*1 - 0 = 2$$
$$z_1 = f(z_{in1}) = 1$$
$$z_{in2} = 2x_2 - x_1 = 2*0 - 1 = -1$$
$$z_2 = f(z_{in2}) = 0$$
$$z_{in} = 2z_1 + z_2 = 2*1 + 2*0 = 2.$$
$$y = f(y_{in}) = 1$$

Example 3.7 Generate ANDNOT function using McCulloch-Pitts neural net by a MATLAB program.

Solution The truth table for the ANDNOT function is as follows:

X_1	X_2	Y
0	0	0
0	1	0
1	0	1
1	1	0

The MATLAB program is given by,

Program

```
%ANDNOT function using Mcculloch-Pitts neuron
clear;
clc;
%Getting weights and threshold value
disp('Enter weights');
w1=input('Weight w1=');
w2=input('weight w2=');
disp('Enter Threshold Value');
theta=input('theta=');
y=[0 0 0 0];
x1=[0 0 1 1];
x2=[0 1 0 1];
z=[0 0 1 0];
con=1;
while con
  zin=x1*w1+x2*w2;
  for i=1:4
     if zin(i)>=theta
        y(i)=1;
     else
        y(i)=0;
     end
  end
  disp('Output of Net');
  disp(y);
  if y==z
     con=0;
  else
     disp('Net is not learning enter another set of weights and Threshold value');
          w1=input('weight w1=');
          w2=input('weight w2=');
          theta=input('theta=');
  end
end
disp('Mcculloch-Pitts Net for ANDNOT function');
disp('Weights of Neuron');
disp(w1);
disp(w2);
disp('Threshold value');
disp(theta);
```