Group Leader : Hassan Ali (20P-0149)

Group Members : Yasir Nawaz (20P-0557)

Project Name: Password Security System

Supervised By : Sir Shakir ullah Shah

Course : Digital Logic Design (EE227)

Department: Computer Science

Contents:

Components required:

- Switch (9 pcs)
- D Flip flop(4pcs)
- 2-input xnor gate (4 pcs)
- And gate (3 pcs)
- Led bulb (2 pcs)`

And Gate:

An AND gate is a logic gate having two or more inputs and a single output. An AND gate operates on logical multiplication rules. In this gate, if either of the inputs is low (0), then the output is also low. If all of the inputs are high (1), then the output will also be high. An AND gate can have any number of inputs, although 2 input and 3 input AND gates are the most common.

There are two binary digits – 0 and 1. We just told that an AND gate performs a binary multiplication of binary digits. In multiplying 0 with 0 we will get 0, 1 with 0 or 0 with 1 we will get 0. Only we get 1 when 1 is multiplied by 1.

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

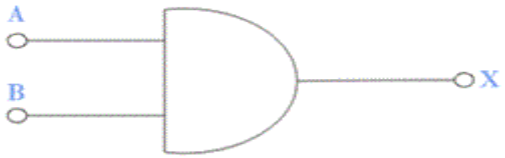
In other words, an AND gate is a digital device that produces high output only when all inputs are high and produces low output under all other inputs conditions. A high digital signal means logically 1 and a low digital signal means logically 0.

Truth Table :

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

2 Input AND Gate

The symbol for a two-input AND gate is logically represented as:



XNOR Gate:

The XNOR gate (also known as a EXNOR or NXOR) – and pronounced as Exclusive NOR – is a digital logic gate whose function is the logical complement of the exclusive OR gate (XOR gate). Logically, an XNOR gate is a NOT gate followed by an XOR gate.

The XOR operation of inputs A and B is $A \oplus B$; therefore, XNOR operation those inputs will be $\overline{(A + B)}$. That means the output of the XOR gate is inverted in the XNOR gate.

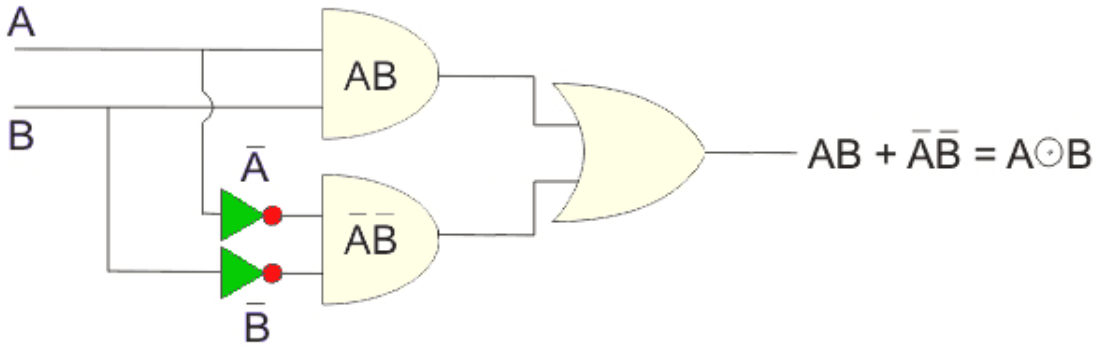
In the XOR gate operation, the output is only 1 when only one input is 1. The output is logical 0 when both inputs are the same, meaning they are either 1 or 0. But in the XNOR gate, the inverse is true. Hence the output is 0 when only one input is 0, and the output is 1 when both inputs are the same (i.e. either two 0's or two 1's).

Truth Table:

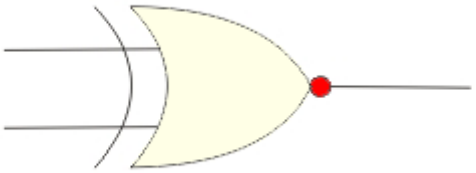
INPUT		OUTPUT
IN1	IN2	OUT
0	0	1
0	1	0
1	0	0
1	1	1

XNOR Gate Circuit Diagram

The expression of XNOR operation can be realized by using two NOT gates, two AND gates, and one OR gate as followers,



The symbol of the XNOR gate:



What is Flip Flop:

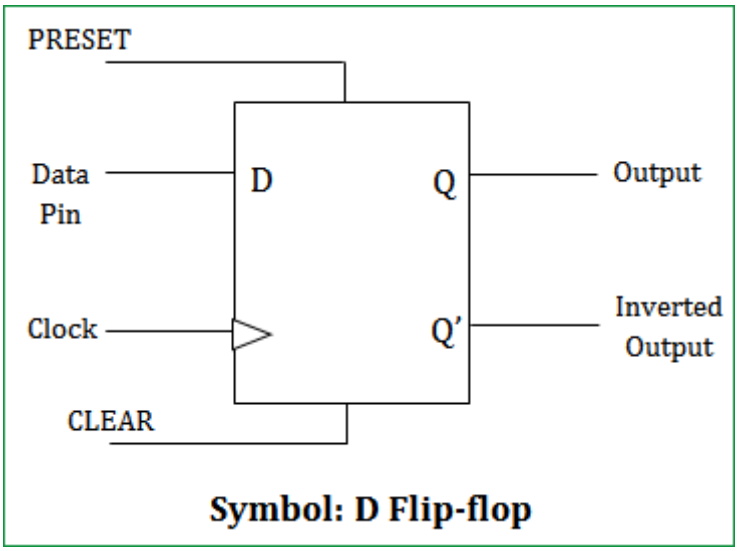
A flip flop is an electronic circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems. Both are used as data storage elements.

D-Flip Flop:

D Flip-flops are used as a part of memory storage elements and data processors as well. D flip-flop can be built using NAND gate or with NOR gate. Due to its versatility they are available as IC packages. The major applications of D flip-flop are to introduce delay in timing circuit, as a buffer, sampling data at specific intervals. D flip-flop is simpler in terms

of wiring connection compared to JK flip-flop. Here we are using NAND gates for demonstrating the D flip flop.

Whenever the clock signal is LOW, the input is never going to affect the output state. The clock has to be high for the inputs to get active. Thus, D flip-flop is a controlled Bi-stable latch where the clock signal is the control signal. Again, this gets divided into positive edge triggered D flip flop and negative edge triggered D flip-flop



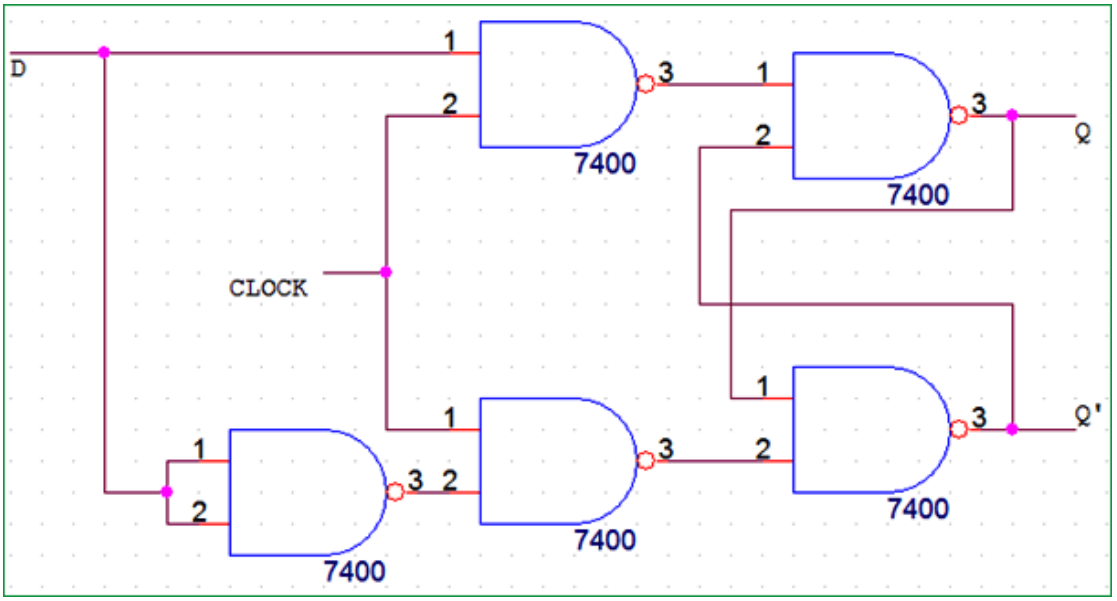
Symbol: D Flip-flop

Truth Table :

Clock	INPUT	OUTPUT	
	D	Q	Q'
LOW	x	0	1
HIGH	0	0	1

HIGH	**1**	**1**	**0**

Representation of D Flip-Flop using Logic Gates:



Software Used:

In this project we used logicly software

Link for downloading and activating logicly:

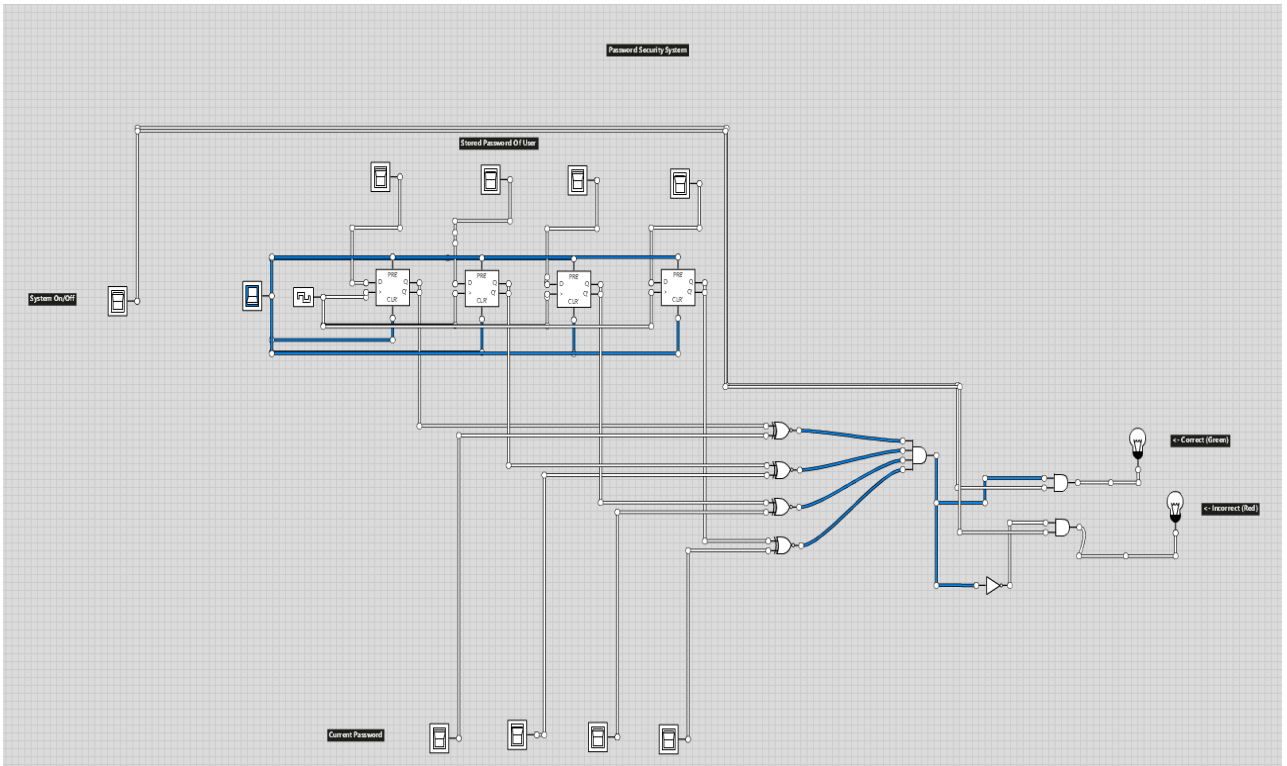
How to Download, Install & Activate Logicly || Logicly Activation Key

Password Security System:

Methodology:

For this project we have used XNOR,FLIP FLOP, SWITCHES, and LEDS. We have two set of input switches one set used for "KEY CODE SWITCHES" switches used to store password and other set of input switches "DATA ENTERY SWITCHES" used for storing current password entered by user. We have stored our password using d flip flop one set of inputs is given to flip flops which stores password other set of inputs is used to get input from user. We have used XNOR gate which will compare bit store in flip flop to corresponding bit given by USER. If both input are same then XNOR will give output 1. Similarly when all 4 inputs will match then all XNOR will give output 1. Now we will use AND gate which will make sure that all input are 1 if yes then output of AND will be 1 . Now using main switch we have two outputs one will be directly connected to LED using AND gate which will compare both output of main AND gate and Main switch. Second output will get an NOT gate attached to it followed by AND get comparing it with main switch and give output to LED bulb which will indicate that PASSWORD DON'T MATCHED.

Circuit Diagram:



Truth Table:

A'	B	B'	C	C'	D	D'	A⊕A'	B⊕B'	C⊕C'	D⊕D'	(A⊕A')'	(B⊕B')'	(C⊕C')'	(D⊕D')'	(A⊕A')' ^ (B⊕B')'	(C⊕C')' ^ (D⊕D')'	((A⊕A')' ^ (B⊕B')') ^ ((C⊕C')' ^ (D⊕D')')	((A⊕A')' ^ (B⊕B')') ^ ((C⊕C')' ^ (D⊕D')')'	KEY	LED GREEN	RED LED
1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	0	1	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	1	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	1	0	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
1	1	0	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	0	1	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	1	0	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	1	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	1	0	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1
0	1	0	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1

A	A'	B	B'	C	C'	D	D'	A⊕A	B⊕B	C⊕C	D⊕D	(A⊕A)'	(B⊕B)'	(C⊕C)'	(D⊕D)'	(A⊕A)' ^ (B⊕B)'	(C⊕C)' ^ (D⊕D)'	((A⊕A)' ^ (B⊕B)') ^ ((C⊕C)' ^ (D⊕D)')	((A⊕A)' ^ (B⊕B)') ^ ((C⊕C)' ^ (D⊕D)')'	KEY	LED GREEN	RED LED
0	1	0	1	0	1	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	0	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	0	1	1	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	1	0	0	1	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	1	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	1	0	1	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
0	1	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	0	1	0	1	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	0	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	0	1	1	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	0	1	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	1	0	0	1	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	1	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	1	0	1	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0
1	0	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0

Logical Expression:

((A0 ⊕ B0)' ^ (A1 ⊕ B1)' ^ (A2 ⊕ B2)' ^ (A3 ⊕ B3)')(Key) ← Green(Correct)

((A0 ⊕ B0)' ^ (A1 ⊕ B1)' ^ (A2 ⊕ B2)' ^ (A3 ⊕ B3)')'(Key) ← Red (Incorrect)