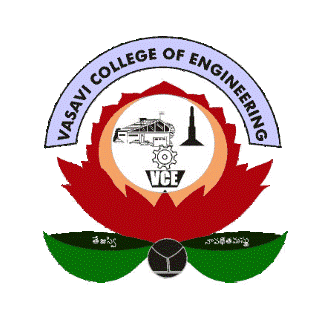
Vasavi college of engineering



**Digital logic design lab 2022/23**

**BE 2/4 3RD SEM**

Lab project report

Mini project: BCD password detector

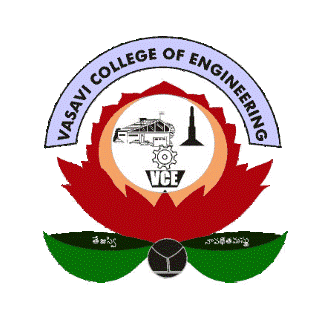
**Submitted by:**

1602-21-735-115(Shivakalyan)

1602-21-735-117(G.Pranav)

1602-21-735-119(Kireeti)

Vasavi college of engineering



**Digital logic design lab 2022/23**

# CERTIFICATE

This is to certify that Shivakalyan,pranav and kireeti had successfully completed the project on "BCD Password Detector" in the Digital Logic Design (DLD) Lab. The project demonstrates their ability to design and implement a password detection system using Binary Coded Decimal (BCD) conversion.

The project was completed with professionalism and efficiency, and the results were thoroughly tested and verified.

The DLD Lab is proud to acknowledge their accomplishment and congratulates them on this achievement.

Signed, [Srikanth sir]

ECE,

Vasavi college of engineering.

# Content:

* Introduction
* What is an FSM?
* Several types of sequence detector
* Password detectors
* Verilog code on BCD password detector
* Explanation of code
* Result
* References

# Introduction:

Let us start with the fundamental question of which language is used in this project, we have used Verilog HDL language, and what is Verilog HDL? Verilog HDL (Hardware Description Language) is a programming language used to model and describe the behavior and structure of digital systems such as digital circuits and integrated circuits. It is a powerful tool that allows designers to describe the behavior and structure of a digital system at different levels of abstraction. Verilog is widely used in the field of digital design, including fields like ASIC and FPGA design.

With Verilog, designers can design and simulate a wide range of digital systems, such as:

1. Digital logic circuits: including basic gates, flip-flops, counters, and state machines.
2. Digital communication systems: such as modems, error-correcting codes, and digital filters.
3. Digital signal processing systems: including filters, transformers, and signal generators.
4. Microprocessors and microcontrollers: such as simple CPUs and controllers for embedded systems.
5. Memory devices: such as RAM and ROM memory.
6. Complex digital systems: such as video decoders and encoders, audio processors, and networking devices.

In summary, Verilog HDL is a powerful programming language used to model and describe digital systems, it allows designers to design and simulate a wide range of digital systems including digital logic circuits, digital communication systems, digital signal processing systems, microprocessors, memory devices, and complex digital systems.

# What is a sequence detector?

A Finite State Machine (FSM) is a mathematical model that describes a system with a finite number of states and transitions between those states. The states represent the different conditions or modes of operation of the system, and the transitions represent the events or inputs that cause the system to change from one state to another.

FSMs are widely used in digital systems design, to model the behavior of digital circuits and to control the flow of data in digital systems.

In summary, FSMs are mathematical models that describe a system with a finite number of states and transitions between those states, they are widely used in digital systems design to model the behavior of digital circuits and control the flow of data in digital systems.

FSMs can be classified into two types: Moore and Mealy. Moore FSM has outputs that depend only on the current state, while Mealy FSM has outputs that depend on both the current state and the inputs.

# Different types of FSM:

FSMs can be classified into two types: Moore and Mealy.

A Moore FSM has outputs that depend only on the current state. In this type of FSM, the output is decided by the state the machine is currently in, and it does not change when transitioning between states.

A Mealy FSM has outputs that depend on both the current state and the inputs. In this type of FSM, the output can change as soon as the inputs change, regardless of the current state.

Both types of FSMs have their own advantages and disadvantages, and the choice of which one to use depends on the specific requirements of the system being designed.

In summary, FSMs are mathematical models that describe a system with a finite number of states and transitions between those states, they are widely used in digital systems design to model the behavior of digital circuits and control the flow of data in digital systems. FSMs can be classified into two types: Moore and Mealy. Moore FSM has outputs that depend only on the current state, while Mealy FSM has outputs that depend on both the current state and the inputs.

# Password detectors:

A password detector is a digital system that is used to detect a specific sequence of input symbols or digits. It is typically implemented using a finite state machine (FSM) and is commonly used in various applications such as access control systems, security systems, and other systems that require a password for authentication.

Password detectors work by accepting an input sequence and comparing it to a predefined password. If the input sequence matches the password, the system unlocks or grants access. If the input sequence does not match the password, the system keeps track of the number of wrong attempts and may lock the system or take other security measures after a certain number of attempts.

Verilog HDL can be used to design and model a password detector by implementing a finite state machine that accepts the input sequence and compares it to the predefined password. The state machine can keep track of the number of wrong attempts and change the output accordingly. Additionally, the code can be optimized to reduce the number of states in the state machine and improve the overall performance of the module.

In summary, password detector is a digital system that is used to detect a specific sequence of input symbols or digits, it can be implemented using a finite state machine (FSM) and is commonly used in various applications such as access control systems, security systems, and other systems that require a password for authentication.

Verilog HDL can be used to design and model a password detector by implementing a FSM that accepts the input sequence and compares it to the predefined password, it can also be optimized to reduce the number of states in the state machine and improve the overall performance of the module.

# Verilog code for BCD password detector:

module pswd\_dtctrdup(passin,rst,nwa,lock,clk,ws,cnt,sat);

input clk,rst;

input [3:0]passin,nwa;

output lock,ws,sat;

output [3:0]cnt;

parameter a=0,b=1,c=2,d=3,e=4,f=5,g=6,h=7;

reg [3:0]ps,ns,cnt;

reg lock,ws,sat;

//resetting the variables

always@(posedge rst or posedge clk)

begin

if(rst==1)

begin

ps=a;

sat=0;

ws=0;

cnt=0;

end

else

ps<=ns;

end

//state diagram logic

always@(passin or ps or rst)

begin

if(ws==1)

begin

cnt=cnt+1;

end

if(sat==1)

begin

ns=a;

lock=0;

end

if(cnt>=nwa)

begin

ns=a;

sat=1;

lock=0;

end

case(ps)

a:begin

lock=0;

if(passin==1)

begin

ns=b;

ws=0;

end

else

begin

ns=a;

ws=1;

end

end

b:begin

lock=0;

if(passin==2)

begin

ws=0;

ns=c;

end

else

begin

ws=1;

ns=a;

end

end

c:begin

lock=0;

if(passin==3)

begin

ws=0;

ns=d;

end

else

begin

ws=1;

ns=a;

end

end

d:begin

lock=0;

if(passin==4)

begin

ws=0;

ns=e;

end

else

begin

ws=1;

ns=a;

end

end

e:begin

lock=0;

if(passin==5)

begin

ws=0;

ns=f;

end

else

begin

ws=1;

ns=a;

end

end

f:begin

lock=0;

if(passin==6)

begin

ws=0;

ns=g;

end

else

begin

ws=1;

ns=a;

end

end

g:begin

lock=0;

if(passin==7)

begin

ws=0;

ns=h;

end

else

begin

ns=a;

ws=1;

end

end

h:begin

if(passin==8)

begin

lock=1;

ws=0;

ns=a;

end

else

begin

ws=1;

ns=a;

end

end

sat:

if(rst==1)

ns=a;

else

ns=ps;

endcase

end

endmodule

# Working of the above code:

The code defines a module named "pswd\_dtctrdup" with inputs "passin", "rst", "nwa", "clk", and outputs "lock", "ws", "sat", "cnt". The module has 8 input and 7 output parameters.

The first always block is used to reset the variables to their initial value when the reset signal (rst) is triggered or when a clock edge is detected. The 'ps' and 'ns' registers store the current and next state of the state machine, respectively.

The 'cnt' register keeps track of the number of incorrect password attempts. The 'sat' flag indicates if the maximum number of attempts has been exceeded, and the 'ws' flag is used to track if the user is entering the password in the correct sequence.

The second always block contains the state machine logic. The input "passin" is compared against the expected values in each state. If the correct value is entered, the state machine transitions to the next state.

If an incorrect value is entered, the 'ws' flag is set, and the state machine returns to the initial state 'a'. The 'lock' output is set to 1 when the state machine reaches the final state 'h', indicating that the correct password has been entered.

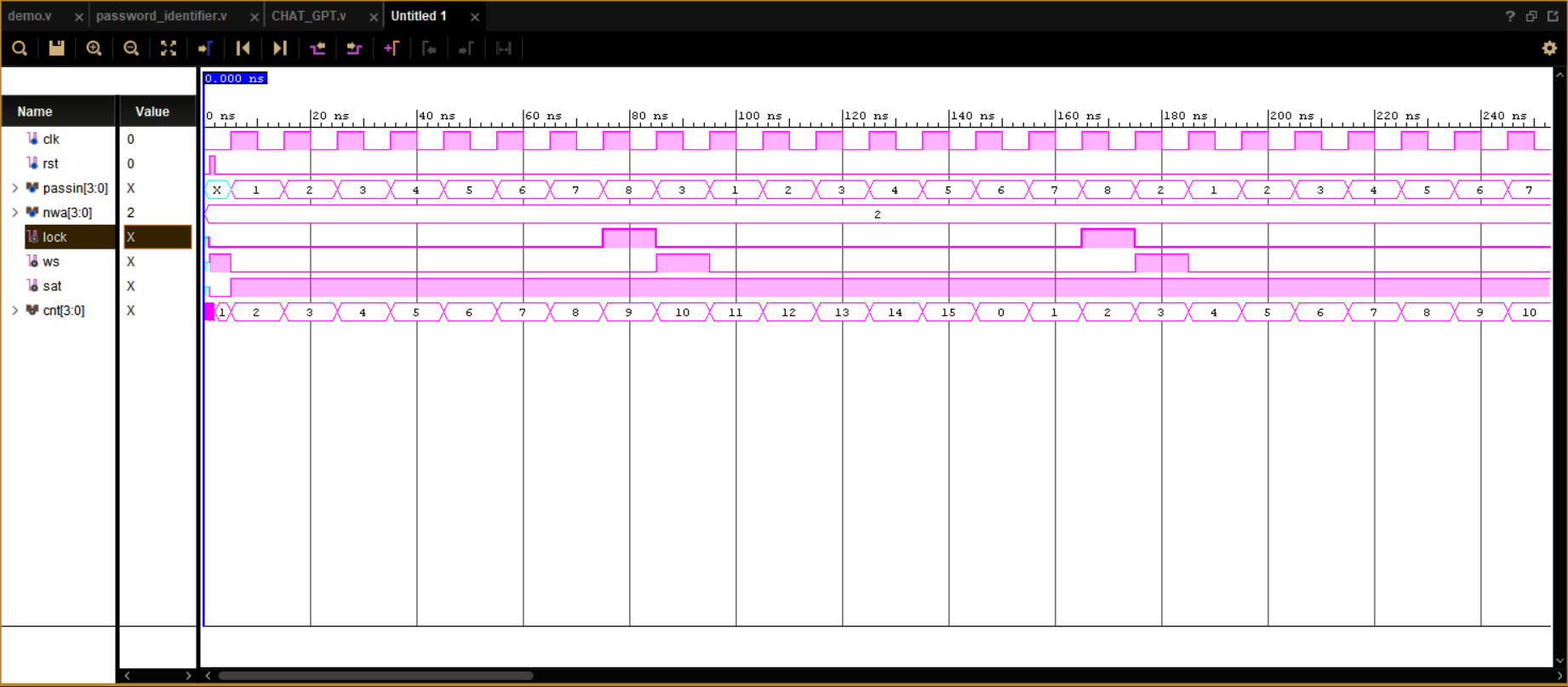
The 'nwa' input is used to set the maximum number of attempts before the password is locked. The 'cnt' register is incremented every time an incorrect password is entered. If 'cnt' exceeds the value of 'nwa', the 'sat' flag is set, and the state machine is forced back to the initial state.

The parameter values 'a', 'b', 'c', 'd', 'e', 'f', 'g', and 'h' define the 8 states of the state machine. Each state is associated with a specific password value and performs a specific action based on the input 'passin'.

The code uses a case statement to implement the state transition logic. Each state checks the value of 'passin' and updates the 'ns' and 'ws' variables accordingly. If the correct password value is entered, the state machine transitions to the next state. If an incorrect value is entered, the 'ws' flag is set, and the state machine returns to the initial state.

In conclusion, this code implements a password detector and duplicate checker using a state machine in Verilog. It takes a sequence of input values and outputs a lock signal if the correct password is entered. The number of password attempts is limited, and the password is locked if the maximum number of attempts is exceeded.

# Result:



This is the output waveform obtained for the password detector and in this its clearly visible that the lock is getting activated only when the sequence is detected and the rest of times it stays off, when a wrong input is entered immediately the ws is activated indicating the occurrence of wrong state.

# Future Scope:

This project can further employe machine learning techniques to identify the intruder and send an intruder alert on the owners mobile.

It can also be further modified by implementing the same with a fingerprint which increase more privacy and authentication.

# References:

Verilog HDL: A Guide to Digital Design and Synthesis, Second Edition By Samir Palnitkar.

Digital Design With an Introduction to the Verilog HDL, VHDL, and System Verilog Sixth Edition By Pearson.