Q-Implementing door lock system using Verilog.

Ans-

**input-**

1. Password
2. Clock
3. Reset

**Output-**

1. Command for unlocking
2. If incorrect 3 attempts it will on alarm
3. If the door is locked then it will show red light
4. If it is unlocked it will show green light

**Understanding: -**

* It is just a basic application of FSM.
* Like The user will enter a 4 bit or 8 bit or any digit password.

It will contain states like:

1. Ideal (In this state

Red light will be on

The buzzer should be in 0(off)

The condition this locked by default)

1. it will match the digits individually on by one (if the 1st digit will match it will go to next digit and so on if it will not match it will go to the ideal state)
2. Unlocked (In this state
   * + 1. The Door will unlock
       2. The green led will 1(on)
       3. .It will remain open for given period of time or by default for 5min

4.It will move to ideal state again

**approach: -**

**🔧 1. Objective**

To design and simulate a secure password-based digital door lock system using **Finite State Machines (FSM)** and **Verilog HDL**, which includes:

* Password checking logic
* Lock/unlock control
* Alarm triggering after multiple incorrect attempts
* Auto-locking using a timer

**🧱 2. System Architecture**

The system is divided into three main modules:

**✅ a. Password\_checker.v**

* **Inputs**: clk, rst, key\_in, enter, check
* **Outputs**: door\_unlocked, incorrect\_flag
* **Function**: Stores the correct password and compares it with user-entered input. Allows only 3 attempts before triggering an incorrect\_flag.

**✅ b. Fsm\_con.v**

* **Inputs**: clk, rst, key\_in, enter
* **Outputs**: locked, check, red\_light, green\_light, buzzer\_alarm
* **Function**: Implements the FSM to control system behavior across 5 states:
  + IDLE: Waiting state
  + INPUT\_WAIT: Captures password digits
  + CHECKING: Triggers password check
  + UNLOCKED: Door opens if password is correct
  + ALARM: Triggered after multiple wrong attempts

**✅ c. Top\_module.v**

* **Inputs**: clk, rst, key\_in, enter
* **Outputs**: locked, red\_light, green\_light, alarm
* **Function**: Integrates FSM and password checker, and adds an auto-lock **timer** to re-lock the door after a fixed interval (e.g., 5ms in simulation).

**📊 3. Simulation and Testbench**

**✨ Testbench: Top\_module\_tb.v**

* Clock generated at 100MHz (#5)
* Uses task enter\_digit() to simulate key presses
* Validates:
  + ✅ Correct password unlocks the door
  + ⏳ Door locks again after timeout
  + 🚨 Alarm is triggered after multiple incorrect attempts

**🧠 4. Key Features**

| **Feature** | **Description** |
| --- | --- |
| Password Memory | Hardcoded 4-digit password (2,2,3,4) |
| Attempt Counter | 3 attempts allowed, then alarm is triggered |
| FSM-based Design | Modular and scalable state-based logic |
| Auto-lock Timer | Re-locks door after UNLOCK\_TIME cycles |
| LED & Buzzer Outputs | red\_light, green\_light, buzzer\_alarm indicators |

**📝 5. Future Improvements**

* Add password change functionality using admin mode
* Introduce keypad debounce logic
* Implement real-time timer using counters or external RTC
* Integrate with UART or Bluetooth for remote access
* Add LCD/OLED display for status and feedback

**✅ Conclusion**

This modular and FSM-based approach allows for clear state transitions, scalability, and robust control. The use of Verilog testbenches ensures proper simulation and validation before deployment on FPGA hardware.