# Enigma Encryption Game

## Team Illuminati Design Document

May 2019

## Team Members

Ashwini Chaudhari

Parshi Srinidhi

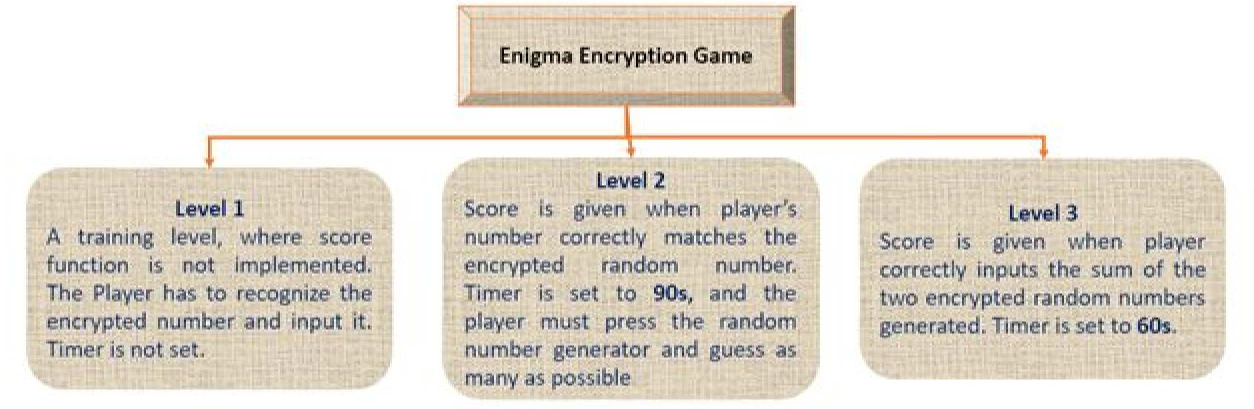
Nikitha Kakani

Manojna Sistla

Prudhvi Thota

Divya Sirikonda

## Game Description


Enigma Encryption Game
Level 1
A training level, where score function is not implemented. The Player has to recognize the encrypted number and input it. Timer is not set.
Level 2
Score is given when player's number correctly matches the encrypted random number. Timer is set to 90s, and the player must press the random number generator and guess as many as possible
Level 3
Score is given when player correctly inputs the sum of the two encrypted random numbers generated. Timer is set to 60s.

This Enigma Encryption game is a multi-user (four users and one guest) authenticated random number encrypted game with timer control. Verilog description of the game is written in Modelsim and debugged using Quartus Prime and implemented on Altera DE-2 115 board.

The game consists of three levels. The first level is a training level. First, the player must enter the user ID and press the button to verify it. If it's correct, then green led turn on or else red led will turn on, and if the user ID is verified then the player must enter the password and press the button to authenticate the game. if the password is correct then the game is accessed by indicating through a green led. Once the game is accessed level must be set by using the toggle switches and pressing the game button.

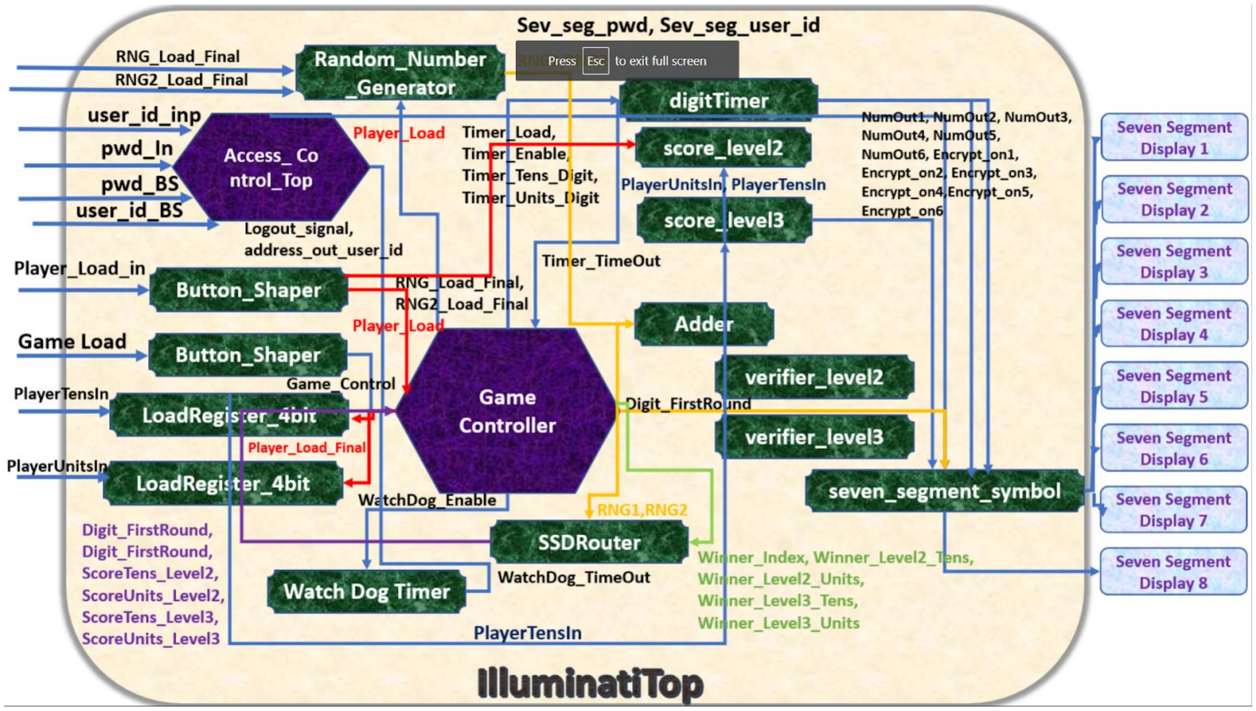
In the first level, the player should identify the encrypted number and the corresponding input and should get familiarized with it. For each button push, encrypted number and its corresponding input number is displayed. The encrypted numbers are from 0 to 9. After completion of this training, a button push makes the game to shift to the second level.

In the second level, a random encrypted number is generated, and the player needs to identify it by matching it with corresponding input in the given time. The timer is set to 90s. When a button is pushed a random encrypted number is generated from 0-9. Player needs to identify that number and give corresponding input from a set of toggle switches and push a button. If the input matches the encrypted number, then the score is updated and green led turns on indicating a correct guess. If the player guesses wrong, then red led turns on and the score is not updated.  The player can play any number of rounds in the given time.  After the time out, the game enters the third level.

In the third level, two random encrypted numbers are generated, and the player needs to input the sum of those two encrypted number and enter through toggle switches in the given time. The timer is set to 60s. When a button is pushed a random encrypted number is generated and one more button push generated another encrypted number. Player needs to calculate the sum of both random numbers encrypted numbers and enter it through toggle switches and push the button to verify. If the sum is verified correctly then, the score is updated.

After completion of the game highest score of a player with the player's index is displayed. There are some additional features in the game. if the button is pushed single time level is shifted, if it is pushed two times in a second then the game is paused. If it's pressed three times, the game is logged out. The progress of the game can be retrieved even if it is logged out. Player can again login using user ID and password and play the game from where he stopped.

## System Level Architecture



The top level module of the game is shown in above Figure is positive edge triggered, synchronous system with active low reset and operated on a 50MHz clock.

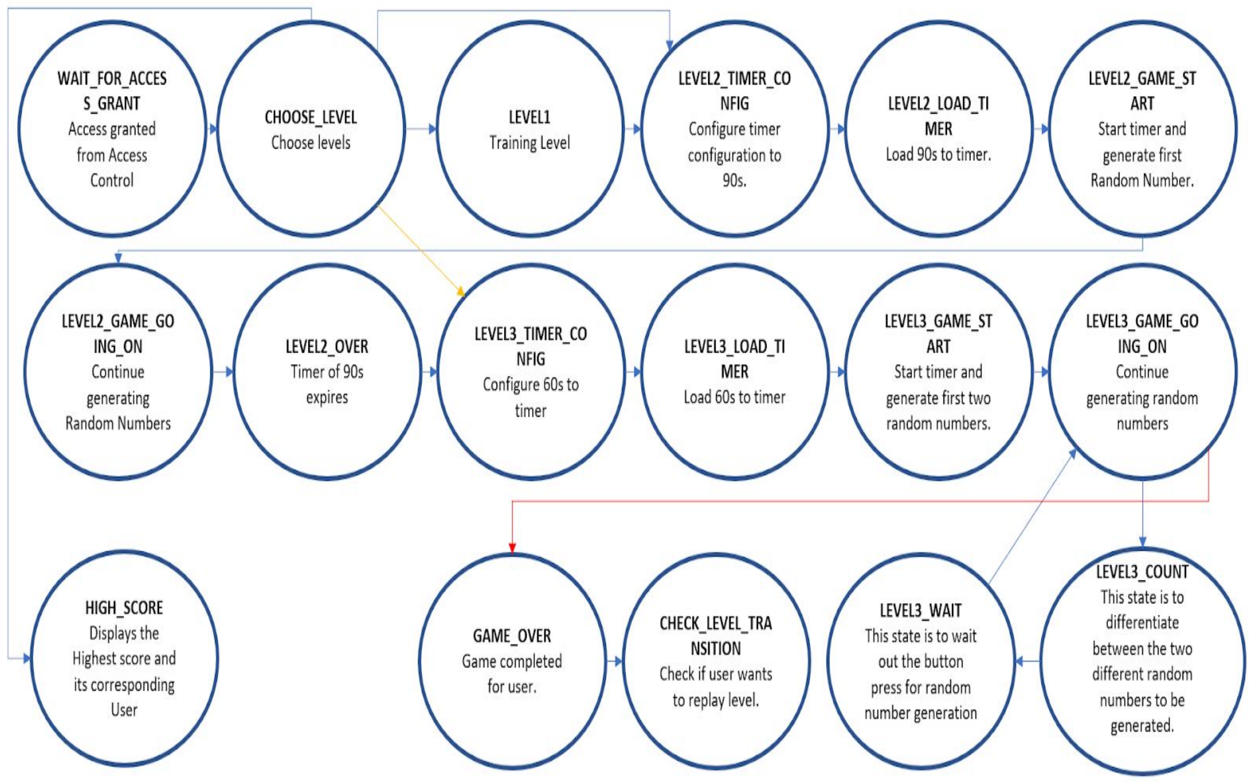
In order to start the game user has to enter his user ID and password using the toggle switches and a button push. The user ID and password authentication is checked by access controller. The entered user ID is matched with the stored user ID in ROM, if it matches then green led glows indicating a valid user ID. Next, player has to enter the password of the corresponding user. If the password matches then a user key is generated and sent to game control for access of game.

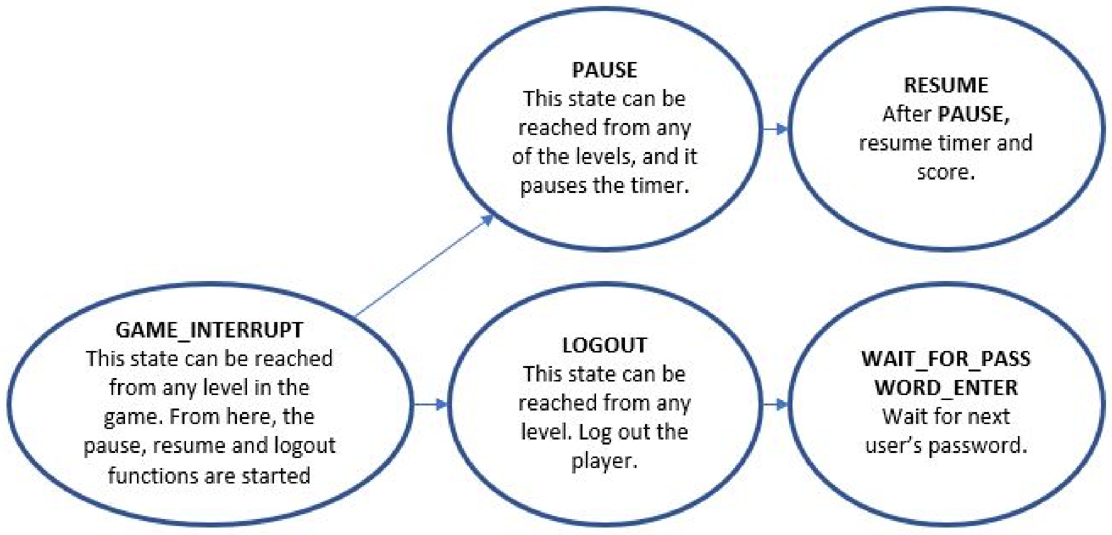
Game controller is the heart of the game. Once the user key is received by game controller it will access the game. Level of game is set using toggle switches. First level is the training level, where user gets trained to encrypted numbers. In the second level, the timer is set to 99 seconds. The timer in this game is generated using Linear Feedback Shift Register (LFSR). In this level, player has to guess the random encrypted number and input the corresponding correct number. Random encrypted number is generated by random number generator module. If the player matches correct, then score is updated. Level 2 score module updates the score for level 2. After completion of level 2 timer, Level 3 is set using toggle switches and a button push. In the level 3 two random encrypted numbers are generated and player has to guess those numbers and input the sum of those two random numbers. The input entered is compared with sum generated by adder module. If the input entered is correct then level 3 score counter updates the score and verifier modules turn on the green led. If input is wrong then verifier module turn on red led. If the button is pressed two times then game timer is paused and another button push resumes game. If the button is pushed three times then game is logged out. At any stage of time the game can be logged out.

## Game Control

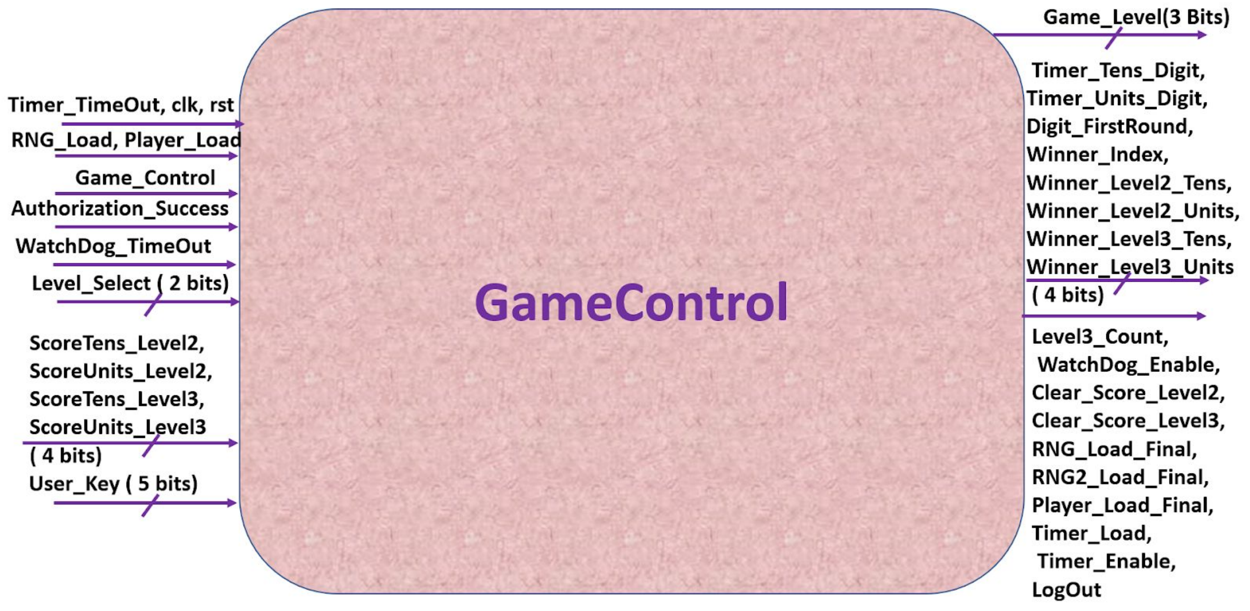
The Game Control module functionality is navigating the different levels in the game and setting out their respective module functions such as Loading timer, incrementing scores, generating random numbers, etc. It has 22 states and complex levels such as level 2&3. It also displays the winner profile at the end of the game. Pause, Resume, and Logout functions are also integrated into the Game Control module as states that can be reached from any Level in the game.

## State Diagrams:


WAIT_FOR_ACCESS_GRANT Access granted from Access Control
CHOOSE_LEVEL Choose levels
LEVEL1 Training Level
LEVEL2_TIMER_CONFIG Configure timer configuration to 90s.
LEVEL2_LOAD_TIMER Load 90s to timer.
LEVEL2_GAME_START Start timer and generate first Random Number.
LEVEL2_GAME_GOING_ON Continue generating Random Numbers
LEVEL2_OVER Timer of 90s expires
LEVEL3_TIMER_CONFIG Configure 60s to timer
LEVEL3_LOAD_TIMER Load 60s to timer
LEVEL3_GAME_START Start timer and generate first two random numbers.
LEVEL3_GAME_GOING_ON Continue generating random numbers
HIGH_SCORE Displays the Highest score and its corresponding User
GAME_OVER Game completed for user.
CHECK_LEVEL_TRANSITION Check if user wants to replay level.
LEVEL3_WAIT This state is to wait out the button press for random number generation
LEVEL3_COUNT This state is to differentiate between the two different random numbers to be generated.


PAUSE This state can be reached from any of the levels, and it pauses the timer.
RESUME After PAUSE, resume timer and score.
GAME_INTERRUPT This state can be reached from any level in the game. From here, the pause, resume and logout functions are started
LOGOUT This state can be reached from any level. Log out the player.
WAIT_FOR_PASSWORD_ENTER Wait for next user's password.

**Block Diagram:**



## Access controller

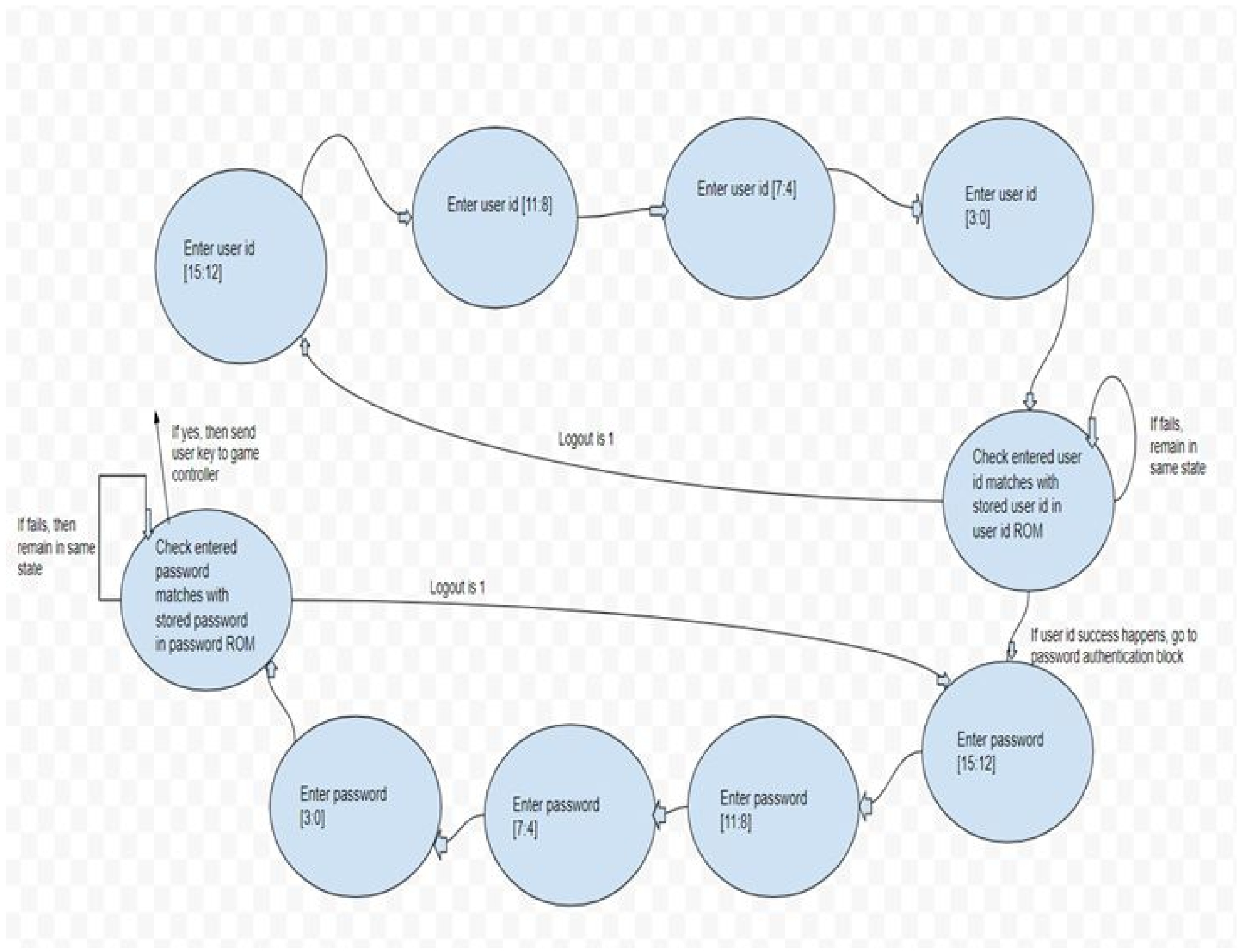
**Access Control unit for multi-user password authentication:**

Access Control unit consist of ROM based user ID authentication module and ROM based password authentication module. It provides password protection to the game. To play the game, player has to enter correct user ID and corresponding correct password. If the password authentication is successful, then player can start with the game, else the player can not start with the game. Access control allows 5 players to play the game. 4 are for the dedicated player and one is one guest player. Both module i.e user ID and password module in access control module does the authentication using ROM.

**Working of Access Control Module:**

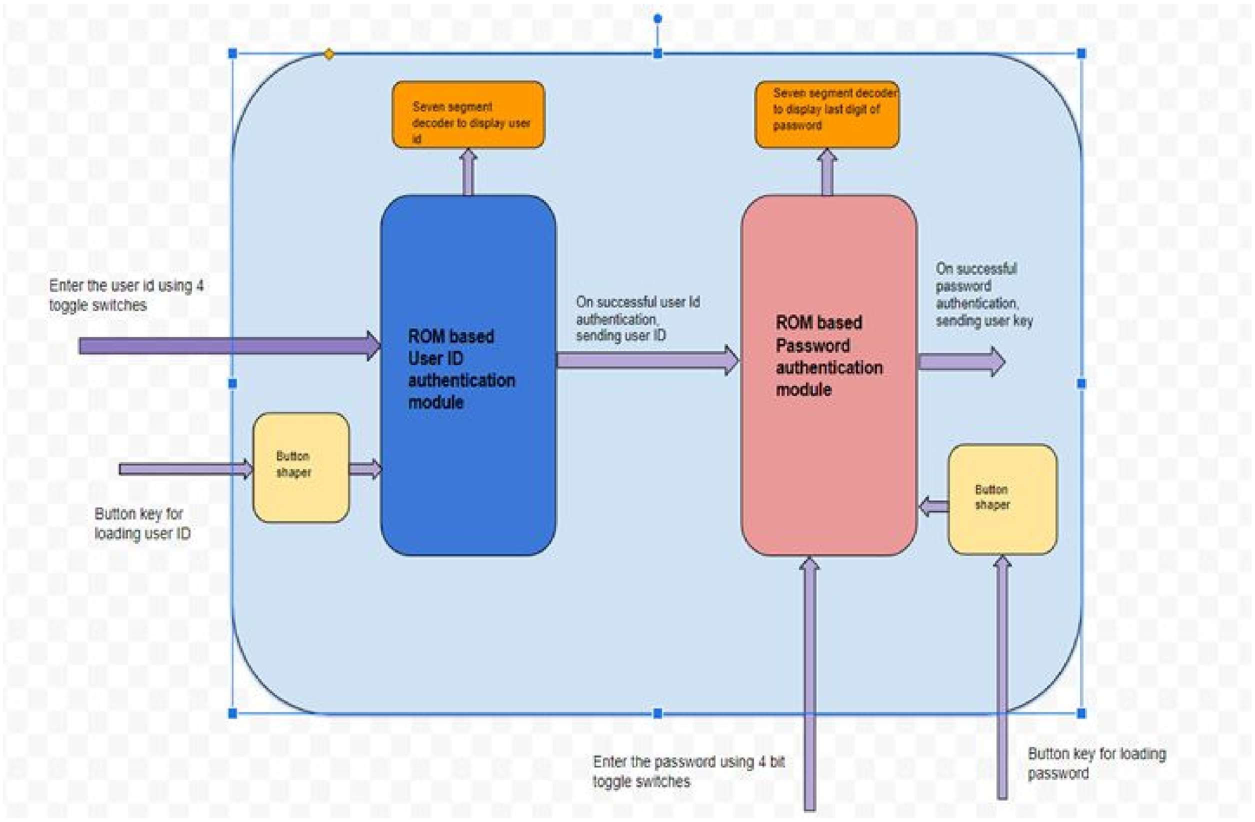
1. Player has to enter the 16 bit user Id with the help of toggle switch one by one and load it with the help of button key. The user ID ROM supports 4 dedicated user IDs and one guest user ID. The entered user ID will be displayed on seven segment display. Also, this entered user ID is compared with each user ID stored in ROM. If it finds match, then it is successful user ID match and the address of that user ID will be sent to Password authentication module.
2. Once the User ID authentication is successful, then green LED will glow indicating that it user ID matches and player can now enter the corresponding correct password. But if user ID authentication fails, then red LED will glow and player cannot start with the game. On successful user ID authentication, the player will enter 16 bit password with the help of 4 toggle switches and load it with help of button key. The last digit of password will be displayed on seven segment display.
3. The entered password will be matched with the corresponding address (sent by user ID module) with ROM. If the entered password matches with stored password, then it is successful password authentication and green LED will glow, else red LED will glow.
4. On successful password authentication, the user key for the player will be sent to game controller module.
5. Also, if the game controller module sends the logout signal on logout condition, both user ID and password authentication module will come back to initial state and player has to start with password and user ID authentication to replay the game.

## Access Control State diagram

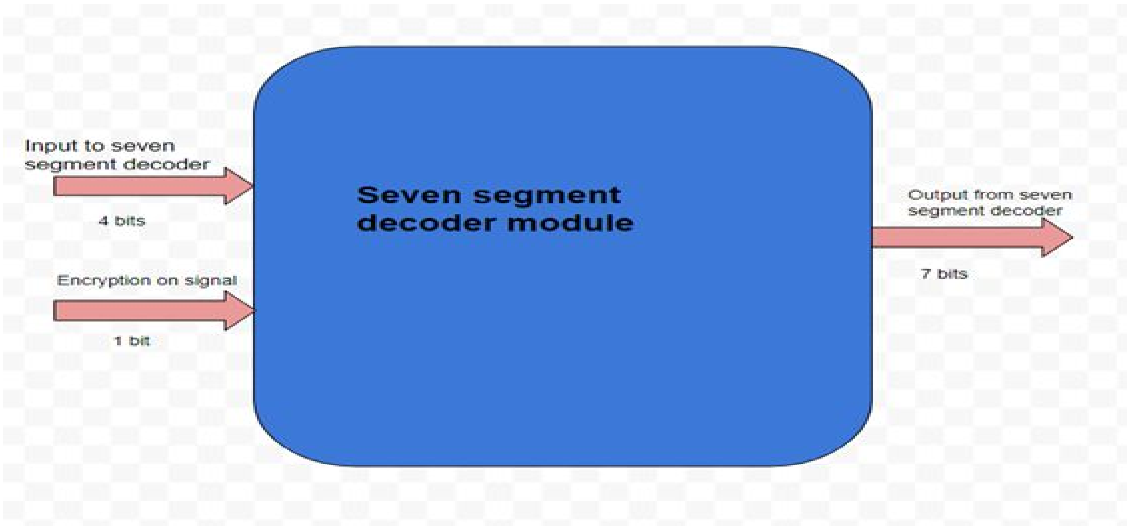


Enter user id [15:12]
Enter user id [11:8]
Enter user id [7:4]
Enter user id [3:0]
Check entered user id matches with stored user id in user id ROM
If fails, remain in same state
Logout is 1
If user id success happens, go to password authentication block
Enter password [15:12]
Enter password [11:8]
Enter password [7:4]
Enter password [3:0]
Check entered password matches with stored password in password ROM
If fails, then remain in same state
If yes, then send user key to game controller
Logout is 1

## Access Control Top module


Seven segment decoder to display user id
Seven segment decoder to display last digit of password
Enter the user id using 4 toggle switches
ROM based User ID authentication module
On successful user Id authentication, sending user ID
ROM based Password authentication module
On successful password authentication, sending user key
Button shaper
Button key for loading user ID
Button shaper
Enter the password using 4 bit toggle switches
Button key for loading password

## Seven segment display

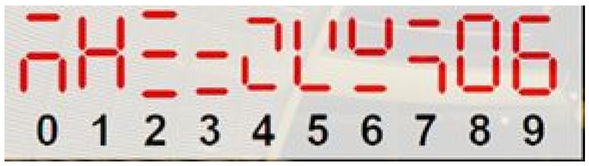

Input to seven segment decoder
4 bits
Encryption on signal
1 bit
Seven segment decoder module
Output from seven segment decoder
7 bits

This module provides the encrypted number/symbol or original number depending on encryption_on signal. Here we are using the number from 0 to 9. This module has two inputs, one is 4 bit input signals which is needed to decode 4 bit input signal to corresponding 7 bit output signal to be displayed on seven segment display.

**Original number**: In original number mode, the encryption_on signal is made to zero and Seven Segment Decoder module will convert the input of 4 bits to a 7 bit output that is feed to the seven segment display. The numbers which are supported to display is from 0 to 9.

**Encrypted number:** In encrypted number mode, the encryption_on signal is made to 1 and seven segment decoder module will convert the 4 bit input signal to corresponding 7 bit encrypted output signal which be feed to seven segment display.

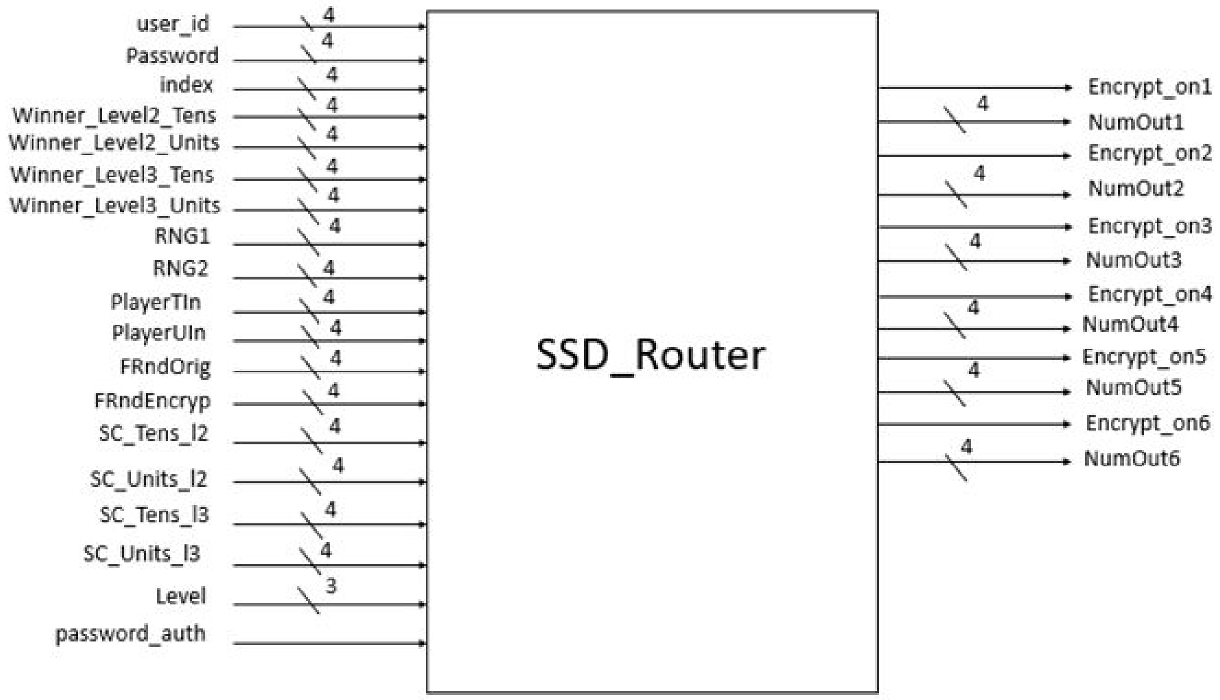
Here you can see the mapping of each digit ranging from 0-9 to encrypted symbol



## SSD Router

Out of the 8 seven segment displays present in the board 6 are overloaded, this module is used to drive the inputs to these 6 displays based on the authentication status and level of the Game. When board is not authenticated then only user ID and Password has to be displayed without encryption. Hence, we route only User_id and password, with Encrypt_on as 0. During the Level 1 of the game, we have to display the last digit of password, The training number and its encryption symbol. Therefore, we drive outputs to 3 of the displays with Encrypt_on as 1 to encryption symbol and 0 to the remaining two. In Level2 as well the last digit of password, Random number, and the input given by the user has to be displayed, along with the score. Encrypt_on will be 1 only for the RNG in this level. For Level 3, the two random numbers and sum entered by user as tens and units digit have to be displayed along with score. In this level, only the score has Encrypt_on as 0 all the other displays show encrypted symbols. When user selects to display High Score the level of the game is set to 4, during this stage, Index of the user with highest score and his scores in Level 2 and Level 3 are displayed. None of them have Encrypt_on as 1. For any other case only last digit of the password is displayed.

Block Diagram:

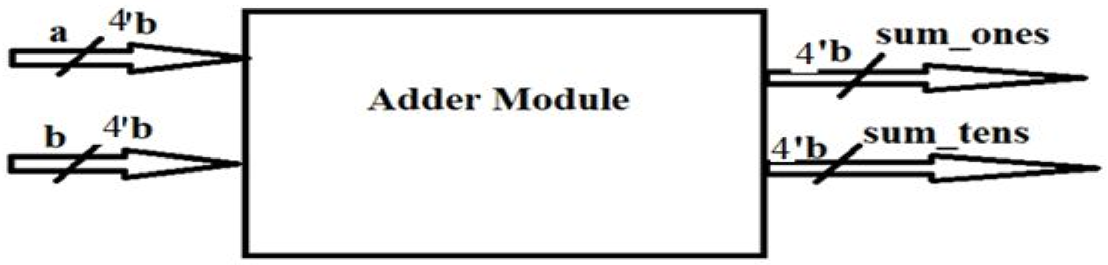


## Adder

The proposed architecture consists of an Adder module. It has two inputs a and b and two outputs one's digit of sum & tens digit of sum. The two numbers are added by using the Adder module and the sum is wired to seven segment display by decoder seven segment module.

Signals used in this module are displayed in below block diagram.

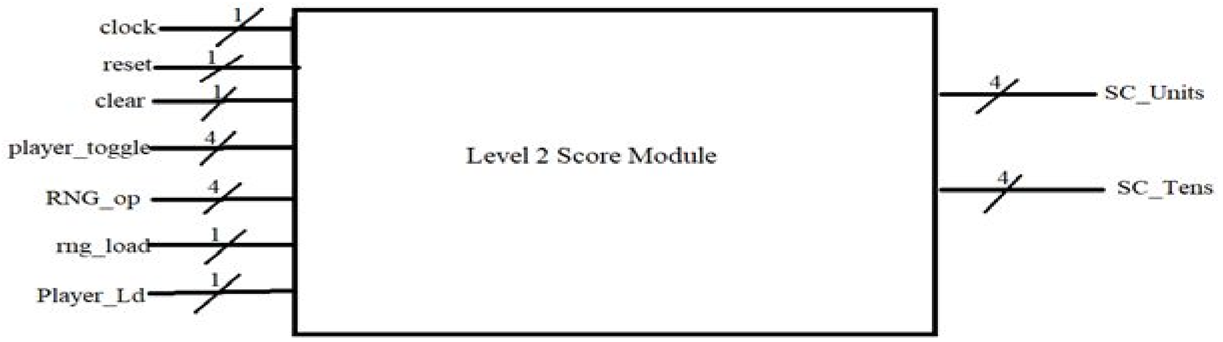
Block Diagram:



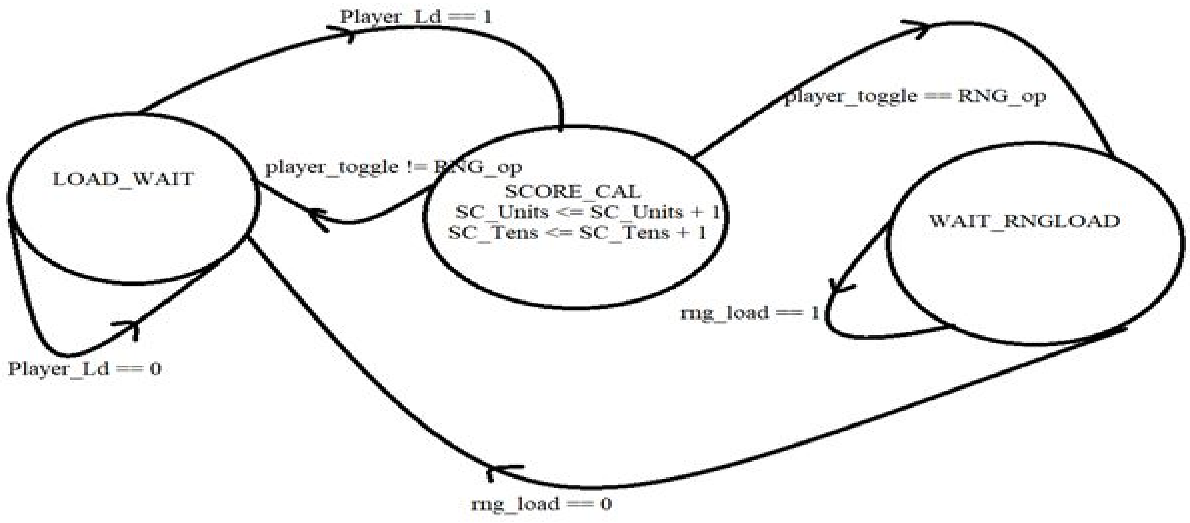
## Level 2 score module

This module consists of inputs such as clock, reset, clear, player_toggle, RNG_op, rng_load, Player_Ld, and outputs are SC_Tens, SC_Units. The main function of this module is when player identified the correct number for generated symbol on seven segment display (comparing player input with the encoded value of symbol) score will be increased by 1 in this level of the game. If he fails to decode the correct value, no points were awarded. The score is displayed in 2 seven segment displays. The 2nd level can be played only when the user completes the 1st level.

Block diagram:



Finite state machine:



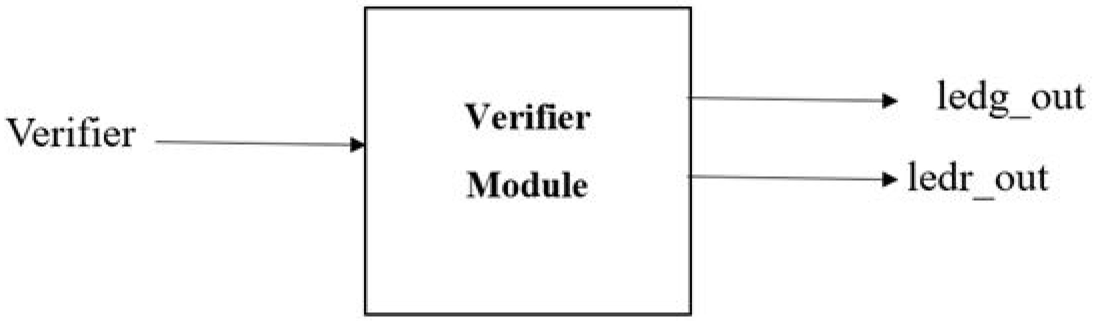
## Verifier module

This module consists of 2 inputs and One output.

Verifier: This signal is high if the player enters the correct sum value in level 3.

ledr_out: Red led signal , This becomes when when verifier is 0 i.e.., the player entered sum value wrong.

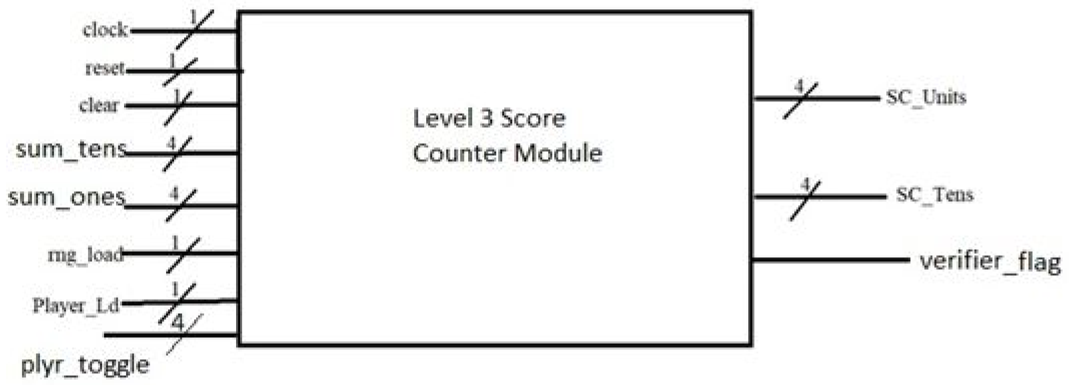
ledg_out:T This LED turns on when the player enters  the correct sum value.



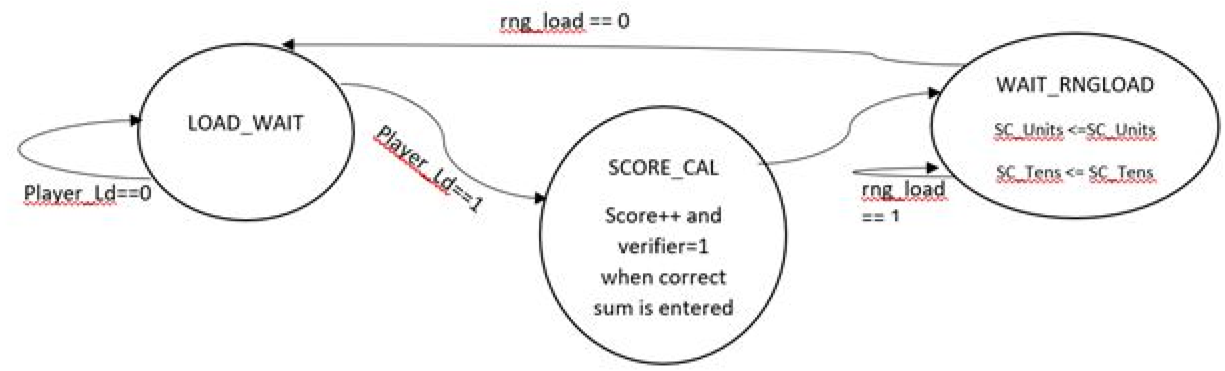
## Level 3 score counter

This module consists of inputs such as clock, reset, clear, plyr_toggle, rng_load, Player_Ld,sum_ones, sum_tens and outputs are SC_Tens, SC_Units, verifier flag. The main function of this module is when player identifies correct numbers for generated symbols on seven segment display and inputted correct sum value of those number then score is increased by one. If he fails to decode correct symbol value and calculate the sum correctly, then there is no increment in the score value. The score is displayed on 2 seven segment displays.

Block diagram:



State diagram:

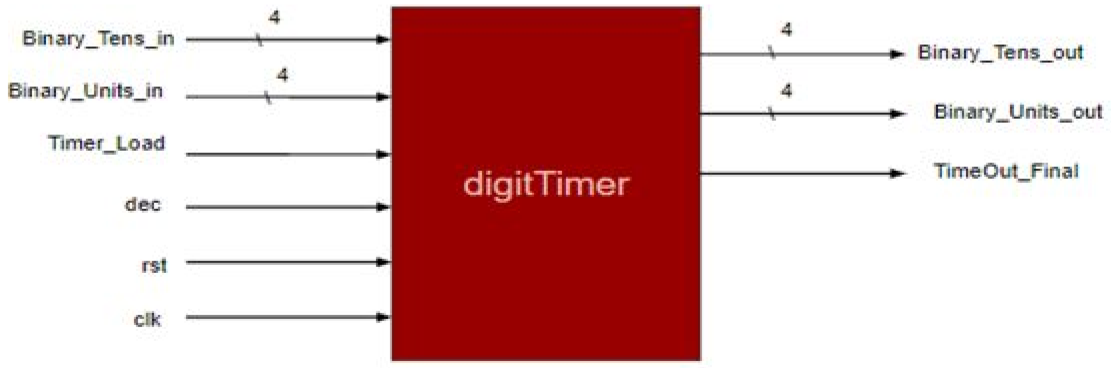


## Digit timer

This module shown in below figure is a sequential circuit which consists of two Digit modules, the input to this module can be given by setting **Binary_in,** and is loaded by giving **Binary_load,** when a decrement(**dec**) pulse is given to the module, the output is decremented, when the output becomes 0 it checks for the **Borrow_disable,** if it is **LOW** then it generates a **Borrow_req** single clock cycle, high pulse and the output becomes 9. If the **Borrow_disable** is **HIGH,** then it generates **TimeOut** signal.

Digit Timer is obtained by instantiating two, digit modules for units and tens place. The output of a one second Timer(which is generated by using counters hierarchically) is given as decrement signal to the timer, The Borrow request signal from units digit is sent as decrement pulse to tens digit and the timeout signal from tens digit is given as borrow disable to the units digit. Borrow disable for tens digit is always HIGH.

Block diagram:

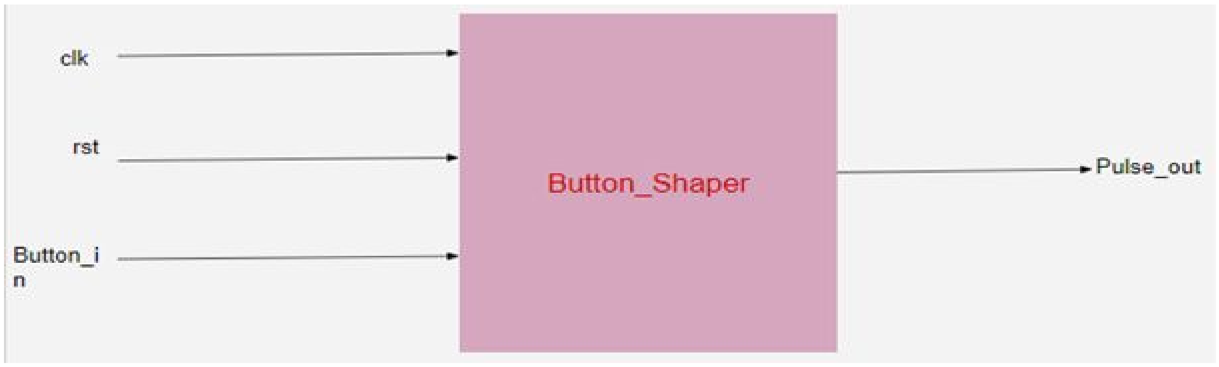


## Button shaper

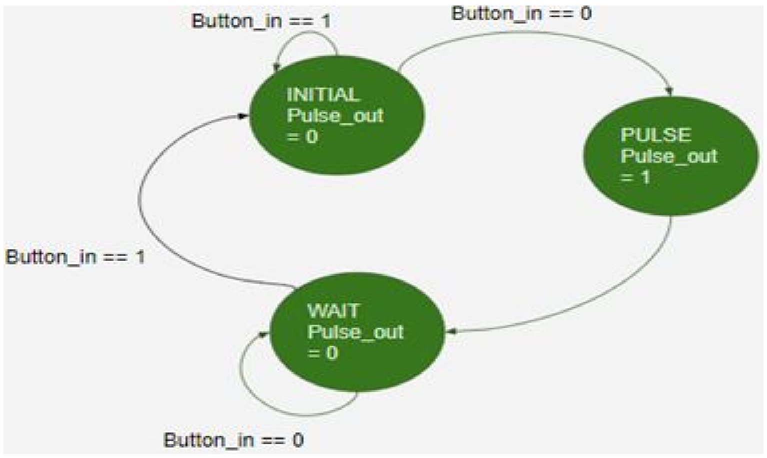
The below figure gives us the overview of Button Shaper module with **clk, rst** and **Button_in** as input signals, the system is positive edge triggered, with Button_in and reset as active low signals, hence whenever a button is pressed the **Button_in** signal becomes **LOW**, in the following positive clock edge the **Pulse_out**(output) becomes **HIGH** and remains in that level only for one clock cycle time, and then becomes **LOW.** This is achieved by considering the input and current state of the system.

The state diagram which helps us obtain the desired output. System will be in **INITIAL** state in the **beginning** and continue to remain in this state till Button is pressed, Once the button is pressed the it goes to **PULSE** stage and the Output becomes **HIGH,** with the positive edge of the clock. In the next clock cycle, the system will go to **WAIT** stage and remains there till any the **input** becomes **HIGH** again. Reset button is used to bring the system to initial stage, that is when **rst** is 0 the system will come to **INITIAL** stage irrespective of the current state.

Block diagram:

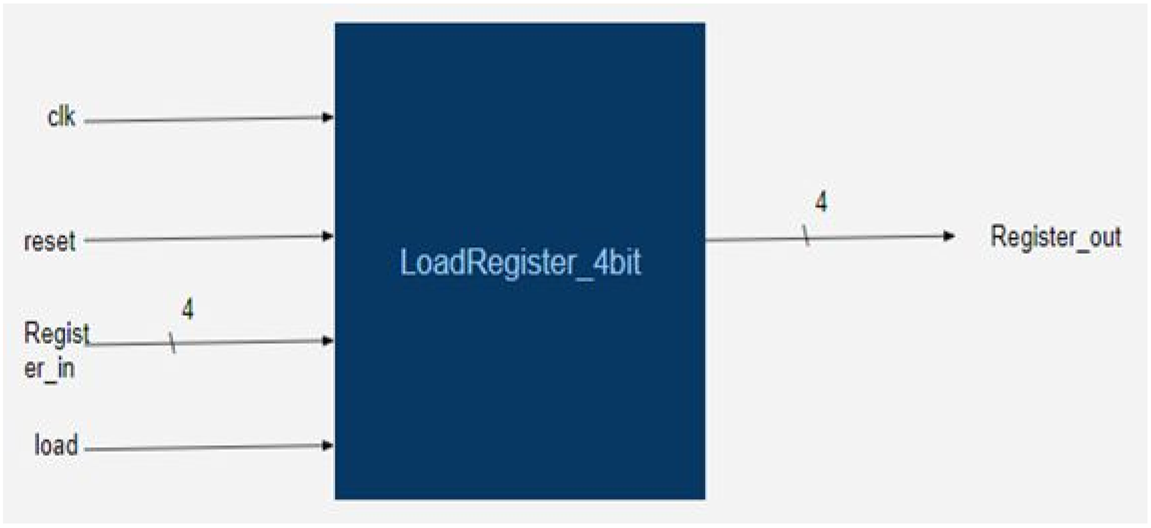


State diagram:



## Load register

The below block diagram is a schematic representation of the Load register, with **clk, reset, load and Register_in** as input signals and **Register_out** as output. When the **load** bit is **HIGH** the input will be driven to output on a positive edge of the clock. If the **load** is **LOW** any change in input will not be reflected in the output, until the load becomes high again. **reset** bit should remain high all the time when it becomes **LOW** all the output bits will be set to 0. Hence, this can be used to clear the output.

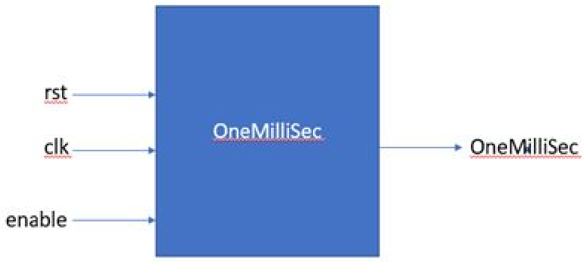


## One second timer

### One milli second

The frequency of clock is 50MHz. So, the time period of each clock is 0.02ms. In-order to get 1ms time we need to multiply each clock time period with 49999.

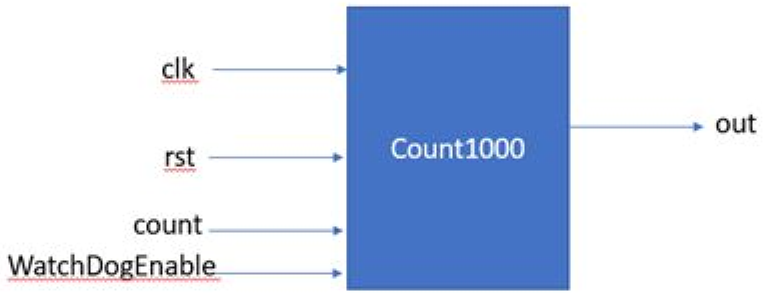
Here we used Linear Feedback Shift Register (LFSR) for generating 1ms time. A LFSR structure is taken and its value at count 49999 is taken. We got its value at count 49999 as 1101101101101100. So, we used this LFSR value to generate 1ms. The timer starts only if enable is 1. At that LFSR value, OneMilliSec output value will be 1 i.e. 1ms time is generated.



**1000 counter**

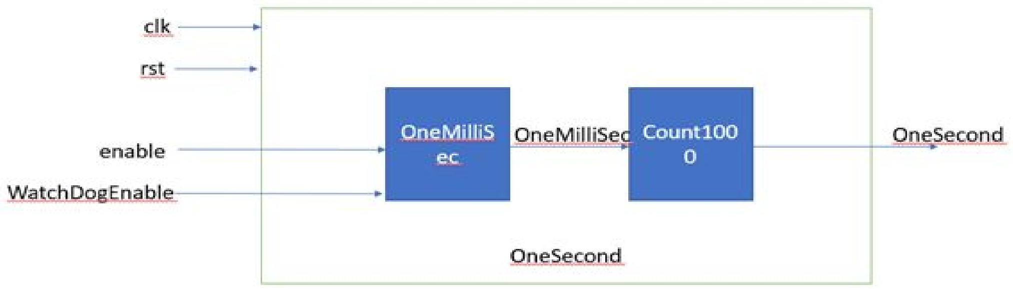
This is 1000 counter. The out will be 1 at the count value 999. This counter is enabled only if count is 1. A WatchDogEnable is signal is give as input. If enable is 1 then, counter starts functioning and if enable is 0 it loses progress and the output becomes 0.

Block diagram:



**One second**

This is one second timer top module. It takes 1ms output and waits till count value is 1000. And then OneSecond output will be 1. This timer starts only if enable and WatchDogEnable are 1. If WatchDogEnable is 0 then, timer stops working.



## Video Description

https://drive.google.com/file/d/1n8acv5HuYbqP9-FvaGrLw5fWjZFxIcQ5/view
In the above video, user ID and password is entered and game is successfully accessed. Level 1 is set and it is the training level where all the random encrypted numbers are displayed. Player got familiarized with these numbers in this level. After completion of level 1, level 2 is set and timer for level 2 is 99 seconds.

https://drive.google.com/file/d/1jtpReTOsdeBM9EWggBVEyhCQoG6yx7oG/view
In second level, player guesses  the random encrypted number and inputted the correct number. As player entered the  correct input, score is updated. For every successful guess, score is getting updated.  Pause and resume functions are also shown.

https://drive.google.com/file/d/1xq9Ijhx7xhxoLfvpW0-VJbeU5U9Ikiay/view
Level 3 is demonstrated in the above video. Two different random encrypted numbers are generated in this level. Player guessed those numbers, calculated sum of those numbers and entered the sum. Highest scorer of the game with corresponding winner index is displayed at the end .