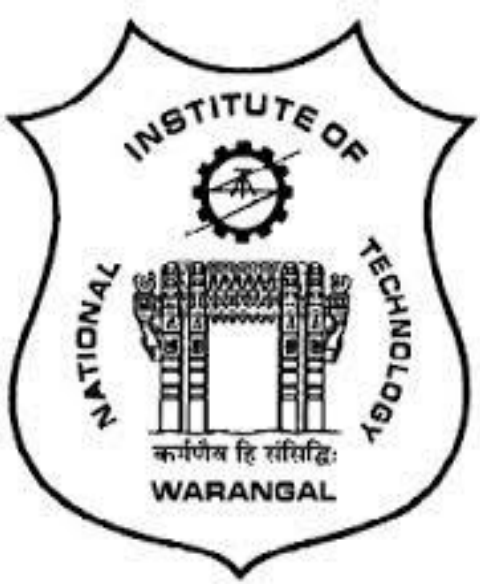


IMPLENTATION OF SAFTEY DEPOSIT VAULT LOCKER ON ARTIX-7 FPGA BOARD

Submitted By

Bhuvana sri vedha vittanala-22ECB0C31
Ramagouni Nisha Priya-22ECB0C33
Vana Meghana-22ECB0C34

Submitted To:
Prof. Ekta Goel(Department of Electronics and Communication Engineering)

EC257- FPGA BASED DESIGN LAB

National Institute of
Technology Warangal 2022-
2026

CONTENT

1.ABSTRACT
2.THEORY
3.VERILOG CODE
4.SIMULATION OUTPUTS
5.IMPLEMENTATION ON ARTIX-7 FPGA BOARD
6.CONCLUSION
7.REFERENCES

ABSTRACT

The aim of this project is to implement a Safety Deposit Vault lock controller using Verilog HDL. A safety deposit vault is an individually owned locker used to safeguard important documents, valuables and so on. It relies heavily on the security level offered by that particular authority. In this specific design, the safety is implemented in various combinations depending on the availability of the owner, bank authorities and the business hours. The protocols are as follows:

During business hours:

Requires an 8-digit code to unlock the safe with the owner of the safe allotted 4-digits, the president 4-digits and the managers' 2-digit code each.

1. The owner of the safe along with the president can access the safe

   _ _ _ _ _ _ _ _

   Owner's 4 digits + president's 4 digits = unlock code

2. The owner of the safe along with both the managers can access the safe.

   _ _ _ _ _ _ _ _

   Owner's 4 digits + manager-1's 2 digit + manager-2's 2 digit = unlock code

Outside of business hours:

Requires a 10-digit code to unlock the safe for added security.

1. The owner of the safe along with the president and one of the managers can access the safe.

   _ _ _ _ _ _ _ _ _

   Owner's 4 digits + president's 4 digits + manager-1's/manager-2's 2 digits

Special case (Non-availability of the owner):

This case arises when the owner of the safe is unavailable due to serious reasons like incapacitation/death of the owner, confiscation of assets etc. then,

1. The president along with both the managers can access the safe along with an override 3-digit access code.

_ _ _ _ _ _ _ _ _

President’s 4 digits + manager-1’s 2 digits + manager-2’s 2 digits

On successful typing in of the access code, override code would be used inaddition. The speciality of this override code is that it is a randomly generated codecreated at the time of inception of the account.

The goal of this project is thus to implement the above system functionally using the building blocks of digital circuits.

THEORY

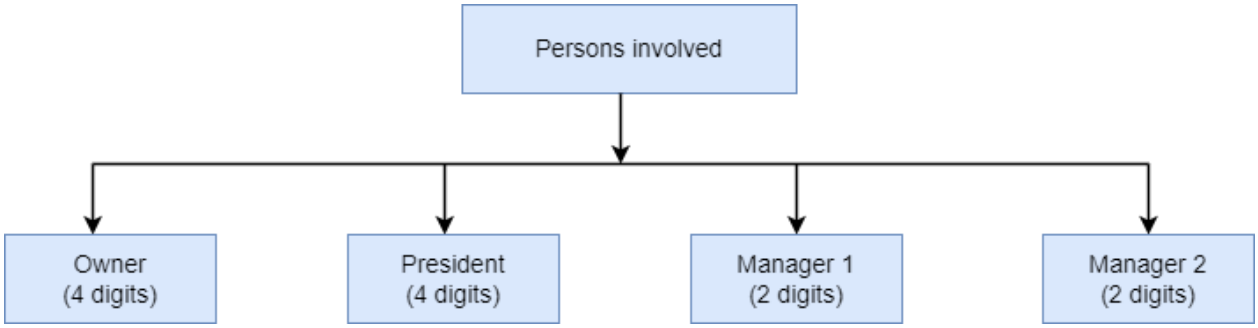
A safe deposit box, also known as a safety deposit box, is an individually secured container, usually held within a larger safe or bank vault. Safe deposit boxes are generally located in banks, post offices or other institutions. Safe deposit boxes are used to store valuable possessions, such as gemstones, precious metals, currency, marketable securities, luxury goods, important documents (e.g. wills, propertydeeds, or birth certificates), or computer data, which need protection from theft, fire, flood, tampering, or other perils. An individual can purchase separate insurance for the safe deposit box in order to cover e.g. theft, fire, flooding or terrorist attacks. Hotels, resorts, and cruise ships sometimes also offer safe deposit boxes or small safes to their patrons, for temporary use during their stay. These facilities may be located behind the reception desk, or securely anchored within private guest rooms for privacy. In the 20th century, bank branches were more prestigious; in the 21st century, space has grown more valuable with higher land values and rents, and many banks see the service as ancillary to their core business.

Most of the safety deposit vaults generally have two locks in the sense that in order to open them, we need two different keys. One of the keys is common to all the safety boxes irrespective of the Owner. The other key hole is reserved to the owner and has a unique signature. The key to the common key hole is held by the Manager of the Safety Deposit Vault,which makes it necessary for the BankManager to be present during the opening of the Locker by the Owner.The present project is basically a digital version of the whole safety deposit vault lock system. Here instead of the keys, we are using passcodes unique to the Owner, the President and the Managers of the Bank. The combinational codes of various possibilities of appearance of the persons would enable the lock to be accessed.However, the President's code is common to all the safety deposit vault

boxes and the only difference being made is the uniqueness of the code of the owner which is decided by the owner himself. Even the Managers are assigned their own code, which is like the President's, common to all the safety deposit vault boxes.However they don’t exercise as much power as either the President or the Owner.

Each person is assigned a specific amount of digits to each code and the various combinations of these codes would then constitute the access code. The number of digits each person is assigned are:

- Owner: 4 digits
- President: 4 digits
- Manager-1: 2 digits
- Manager-2: 2 digits



The ways the safe can be accessed are as follows:

1. Considering that the safety deposit vault is in its banking hours, the presence of the owner and the president would be sufficient. That is, the passcode would comprise 8 digits which is necessarily the confluence of the 4 digits of the owner and 4 digits of the president. Necessarily if the owner’s 4 digits and the president’s 4 digits are typed in correctly, the safe can be accessed.

Point to be noted here is that the president's 4 digits are common to all the other safety deposit vault boxes whereas the owner's is unique to that particular box.

2. Considering that the safety deposit vault is in its banking hours and in the absence of the president, the safe can be accessed by the owner with the help of both the managers being present simultaneously. That is, the passcode would comprise 8 digits which is necessarily the confluence of the 4 digits of the owner, 2 digits of 1 manager and the 2 digits of another manager. Necessarily if the owner's 4 digits and both the managers' 2 digits each are typed in correctly, the safe can be accessed. Another point to be noted is that both the managers' 2 digits are common to all other safety deposit boxes.

3. Now let us consider that the safety deposit vault's official banking hours are done. But then there's some emergency and the owner really needed to access the safe. In this case, the safe can be accessed but with a condition. Instead of the previously said 8 digit code, this time a 10 digit code is needed to access the safe. The 10 digit code is necessarily formed by the combination of owner, president and one of the managers. That is, the 4 digitcode of the owner along with the 4 digit code of the president and the 2 digit code of either one of the managers would form the required 10 digit code.

4. The final and the special case arises when there's a mishap like the cases where the owner dies/ income tax raids, etc. which are necessarily emergency conditions which don't require the presence of the owner. In suchcases, safe can be accessed in the following manner: an 8 digit code needs tobe entered in the first level followed by a 3 digit code. The 8 digit code is

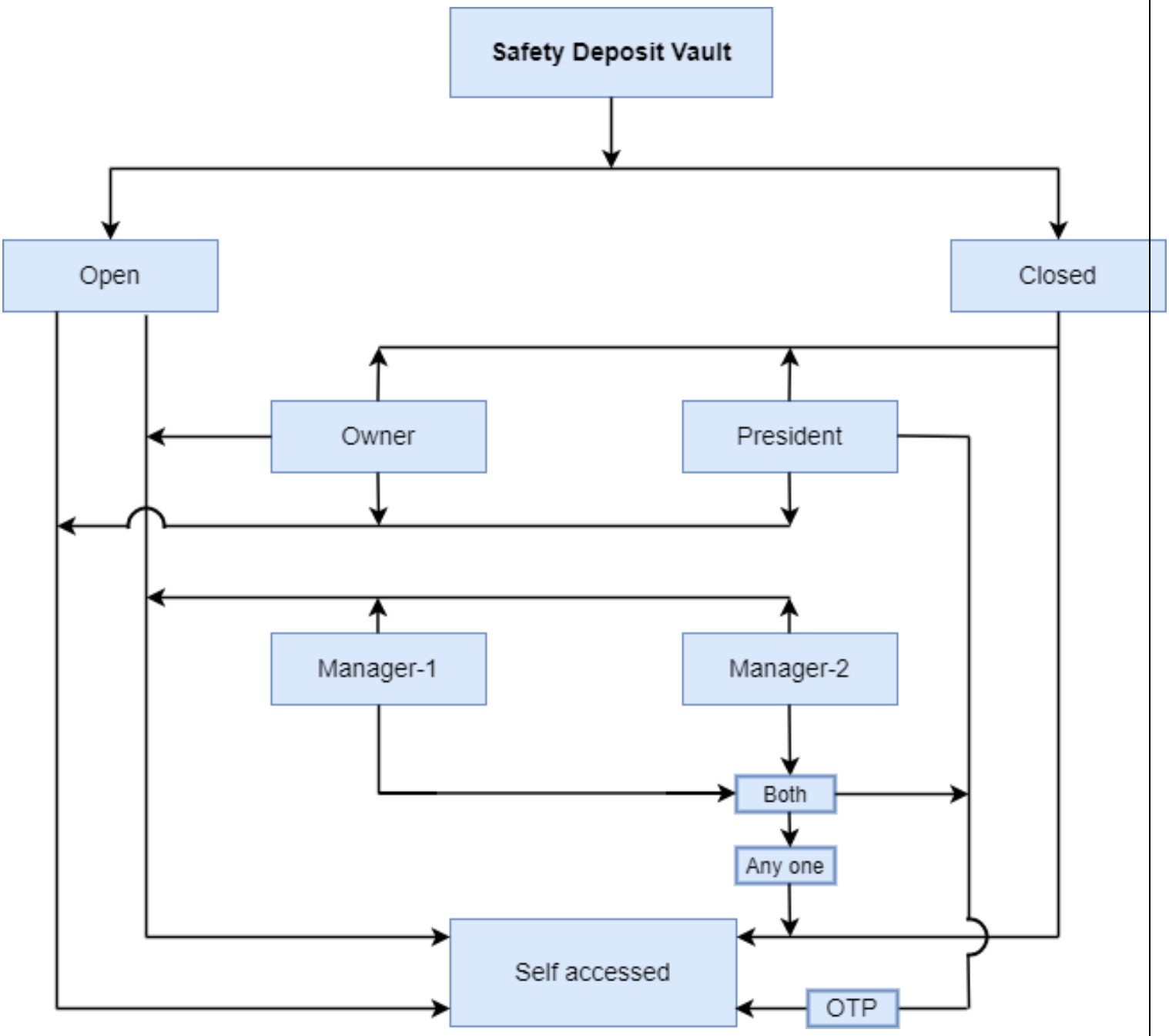
necessarily a confluence of the 4 digits of president, 2+2 digit codes of the managers. These 8 digits together typed in would help to unlock the first level of security.

The 2nd level of security comprises a 3 digit code called as override code. On typing in the correct override code would give access to the safe. This override code is allotted to the user at the opening of the account itself. Unlike the main code digits, this override code is decided by the system.

The override code:

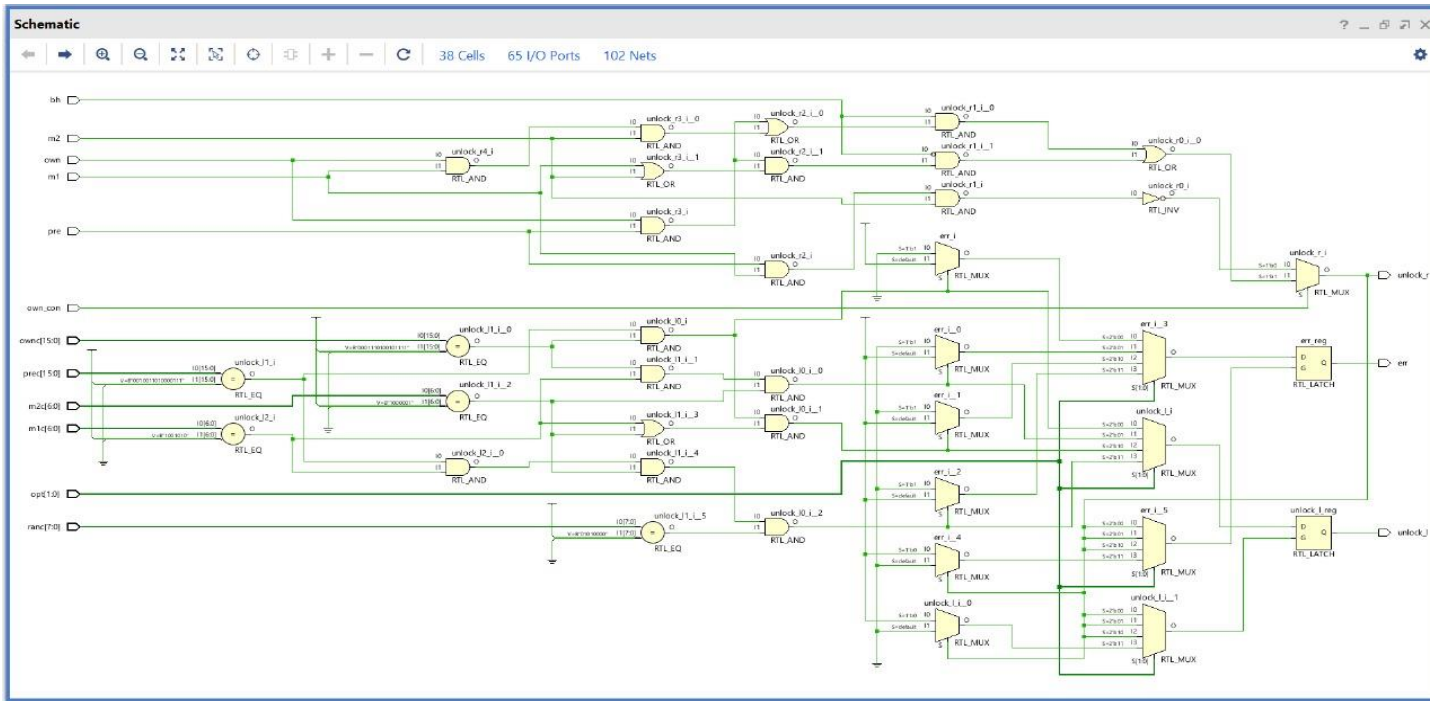
It is a 3 digit code randomly generated by the system at the inception of the account. This randomly generated code is then allotted to each account as an override code which then can be used in dire situations as mentioned above. The random generation of this code has also been implemented.

Design Flow:



Main Module:

Let us now know about the working of the main module.

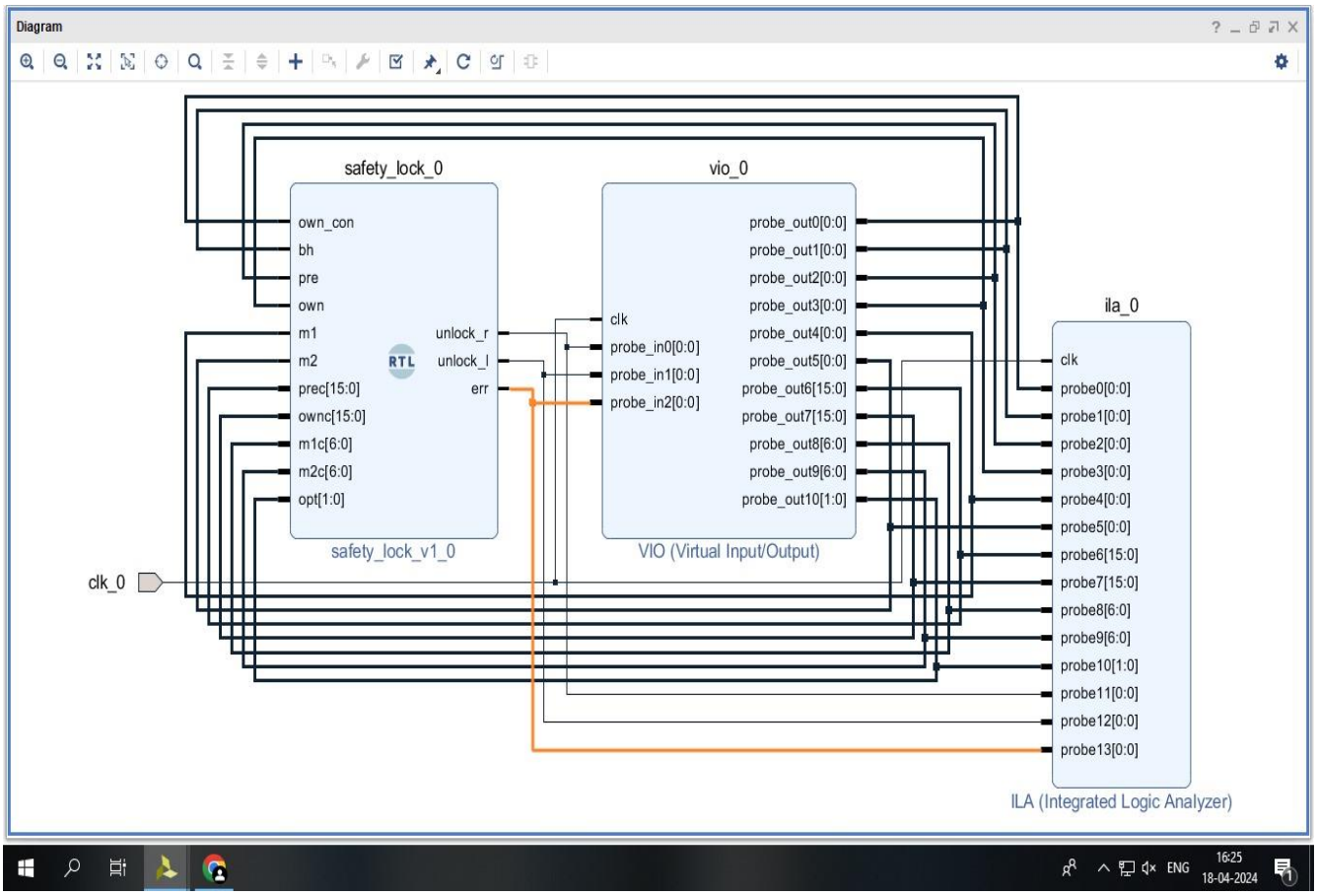


SCHEMATIC DIAGRAM

The above circuit diagram depicts the logic of the whole circuit. Point to be noted here is that the circuit just shows the simple combinations of various cases that are there that help in opening the safe. In each case the presence of each person is verified by the correctness of each pass key entered. Index to variables mentioned in the above circuit diagram are:

- Pre: president
- Own: owner of the safe
- Own_con: owner condition (is 1 if the owner is present and 0 if the owner is absent (in the case 4)).
- Bh: banking hours
- M1: manager 1
- M2: manager 2

The password entered by the user is compared with the password registered in the system which then provides truthfulness to the variables pre, own, m1, m2 which then leads to the opening of the safe. For the first 3 cases, the own_con remains true because the owner is present. Let us now consider the final and special case, i.e. the absence of the owner. Just the president, managers 1 and 2 are required to access the first level of security. The second level of security can only be bypassed by using the override code whose generation is about to be discussed now.



BLOCK DIAGRAM

VERILOG CODE:

```
module safety_lock(own_con, bh, pre, own, m1, m2, opt, unlock_r,
 unlock_l, prec, ownc, m1c, m2c, ranc, err);
input own_con, bh, pre, own, m1, m2;
input [15:0] prec, ownc;
input [6:0]m1c, m2c;
input [1:0] opt;
input [7:0] ranc;
output  reg unlock_r, unlock_l, err;
/*own_con - owner_condition(alive or not),
 bh - banking hours, pre - president, own - owwner,
  m1/2 - manager1/2, opt - option(for virtual keypad),
 unlock_r- unlock_room, unlock_l- unlock_locker, prec- president
  code,
 ownc- owner code, m1/2c- manager1/2 code, err- error signal(if
 incorrect password)*/
 parameter OWNC = 16'd7471; //predefined owner code
parameter PREC = 16'd9863; //predefined president code
parameter M1C = 7'd74; //predefined manager1 code
parameter M2C = 7'd65; //predefined manager2 code
parameter RANC = 8'd80; //predefined random code
// for unlocking the lockers room
always@(pre or own or bh or m1 or m2)
begin
case(own_con)
 1'b0 : begin  unlock_r = ~(pre & m1 & m2);
end
1'b1 : begin  unlock_r = ((bh & ((own & pre) | (own &
m1 & m2))) |
( ~bh & ((own & pre & (m1 | m2)))));
end
endcase
end
// for unlocking the locker
always@(opt)
begin
```

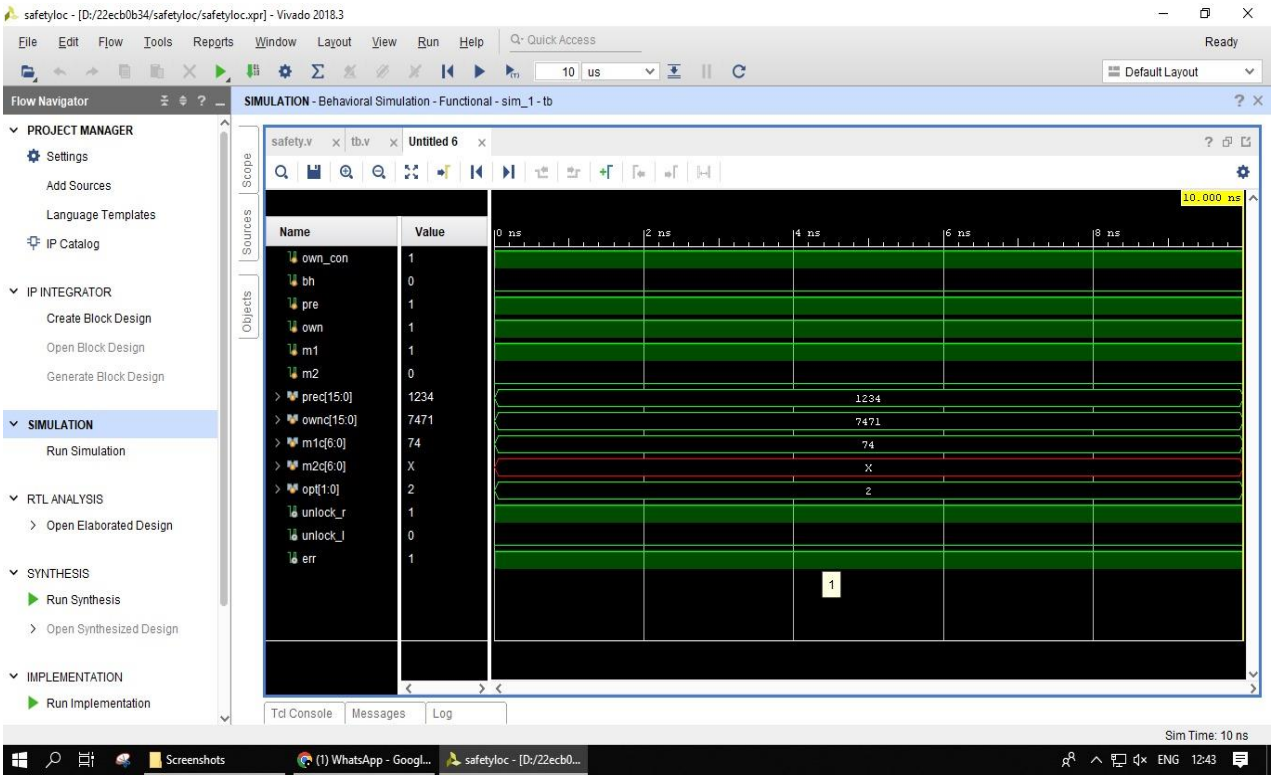
```
 case(opt)
  // bh + pre + own



 2'b00 : if(unlock_r == 1'b1)
 begin
 if((prec == PREC) & (ownc == OWNC)) begin
 unlock_l = 1'b1;
 err=1'b0;
 end
else begin
unlock_l=1'b0;
err = 1'b1;
end
end
// bh + own + m1 + m2
2'b01 : if(unlock_r == 1'b1)
begin
if(ownc == OWNC & m1c == M1C & m2c == M2C)begin
unlock_l = 1'b1;
err=1'b0;
end
else begin
unlock_l=1'b0;
err = 1'b1;
end
end
// bh' +own+ prec+m1|m2
2'b10 : if(unlock_r == 1'b1)
begin
if(ownc == OWNC & prec == PREC & (m1c == M1C|
m2c == M2C)) begin
unlock_l = 1'b1;
err=1'b0;
end
else begin
unlock_l=1'b0;
err = 1'b1;
end
```

```
end
// pre + m1 + m2 + ranc
2'b11 : if(unlock_r == 1'b0)
begin



 if(prec == PREC & m1c == M1C & m2c == M2C & ranc
 == RANC) begin
 unlock_l = 1'b1;
 err=1'b0;
 end
else begin
unlock_l=1'b0;
err = 1'b1;
end
end
endcase
end
endmodule
```

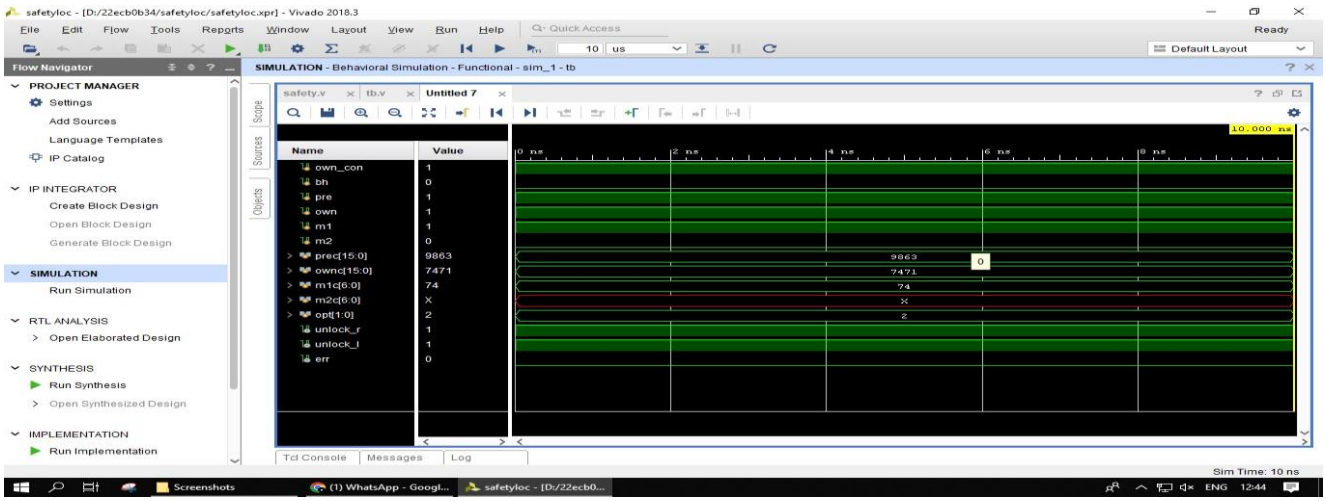
TEST BENCH CODE:

```
module tb;
reg own_con, bh, pre, own, m1, m2;
reg [15:0] prec, ownc;
reg [6:0] m1c, m2c;
reg [7:0] ranc;
reg [1:0] opt;
wire unlock_r, unlock_l, err;
lock uut(own_con, bh, pre, own, m1, m2, opt, unlock_r,
unlock_l, prec, ownc, m1c, m2c, ranc, err);
initial begin
own_con=1'b0;bh=1'b1;own=1'b0;pre=1'b1;m1=1'b1;
m2=1'b1;opt=2'b11;
prec=16'd9863;ownc=16'd7471;m1c=7'd74;m2c=7'd65;
ranc=8'd80;
#30
$finish();
end
endmodule
```

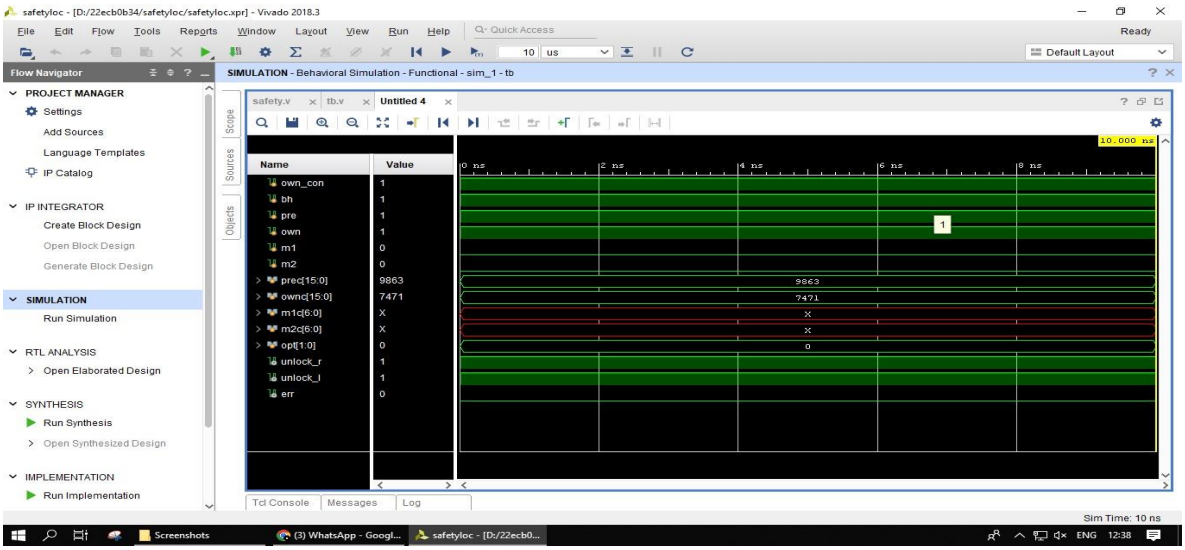
simulation outputs



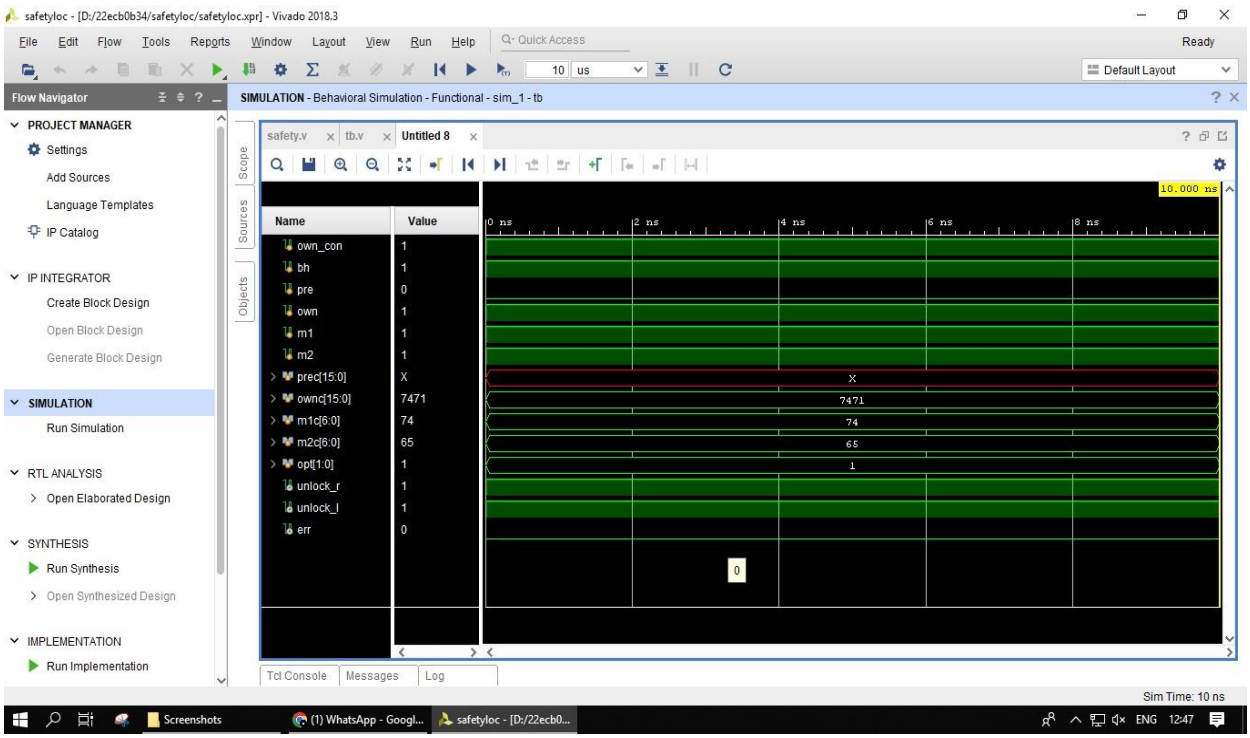
1.BH'+PRESIDENT+OWNER+MANGER1(INCORRECT PASSWORD)



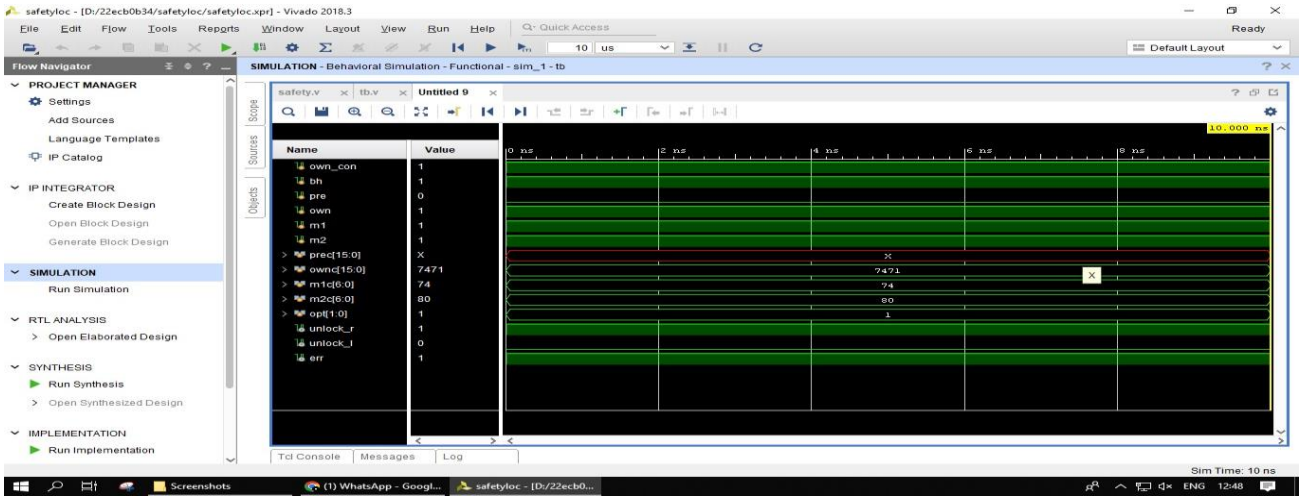
2.BH'+PRESIDENT+OWNER+MANAGER1(CORRECT PASSWORD)



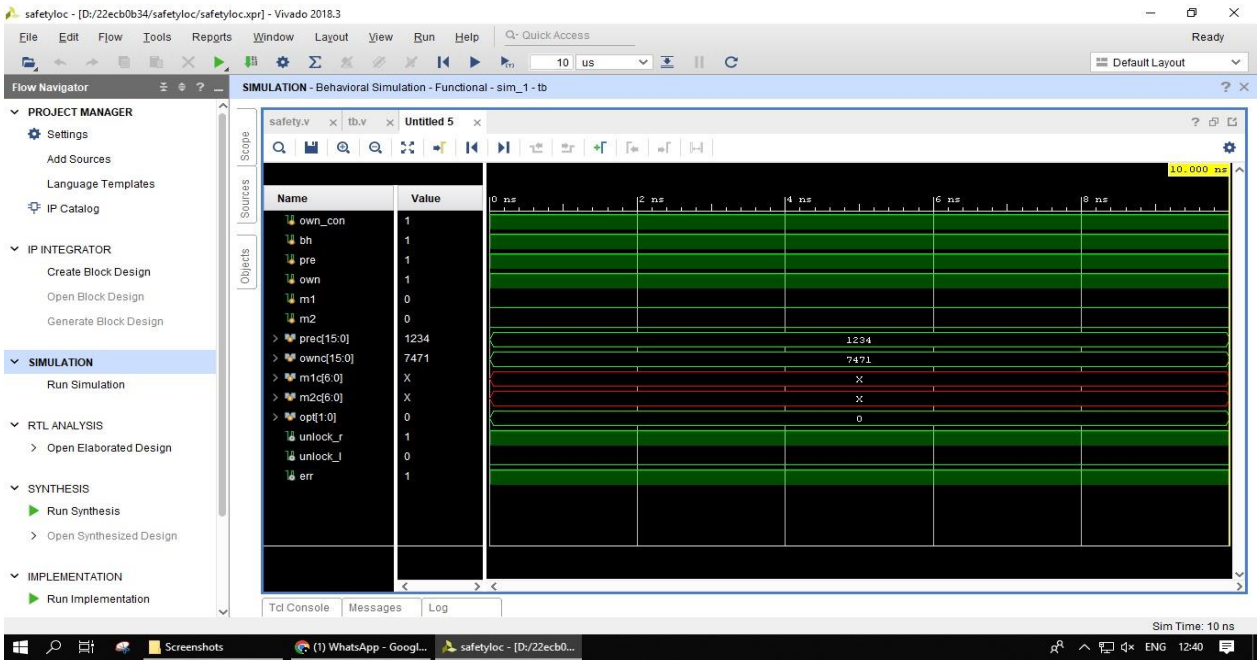
3.BH+OWNER+PRESIDENT (CORRECT PASSWORD)



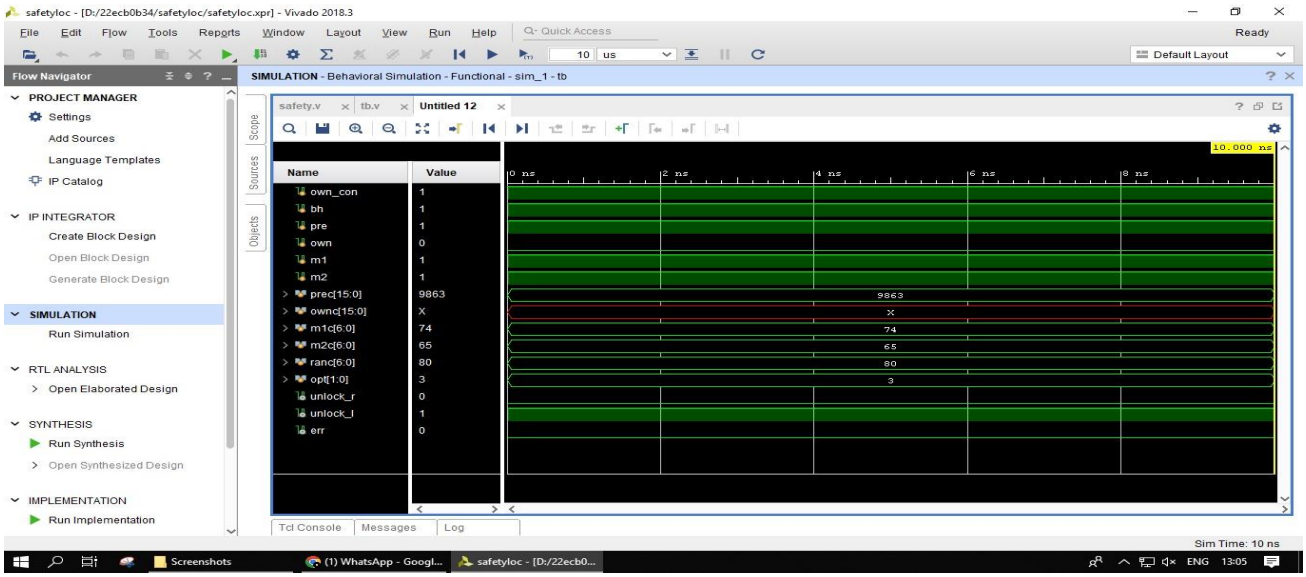
4.BH+OWNER+MANAGER1+MANAGER2(CORRECT PASSWORD)



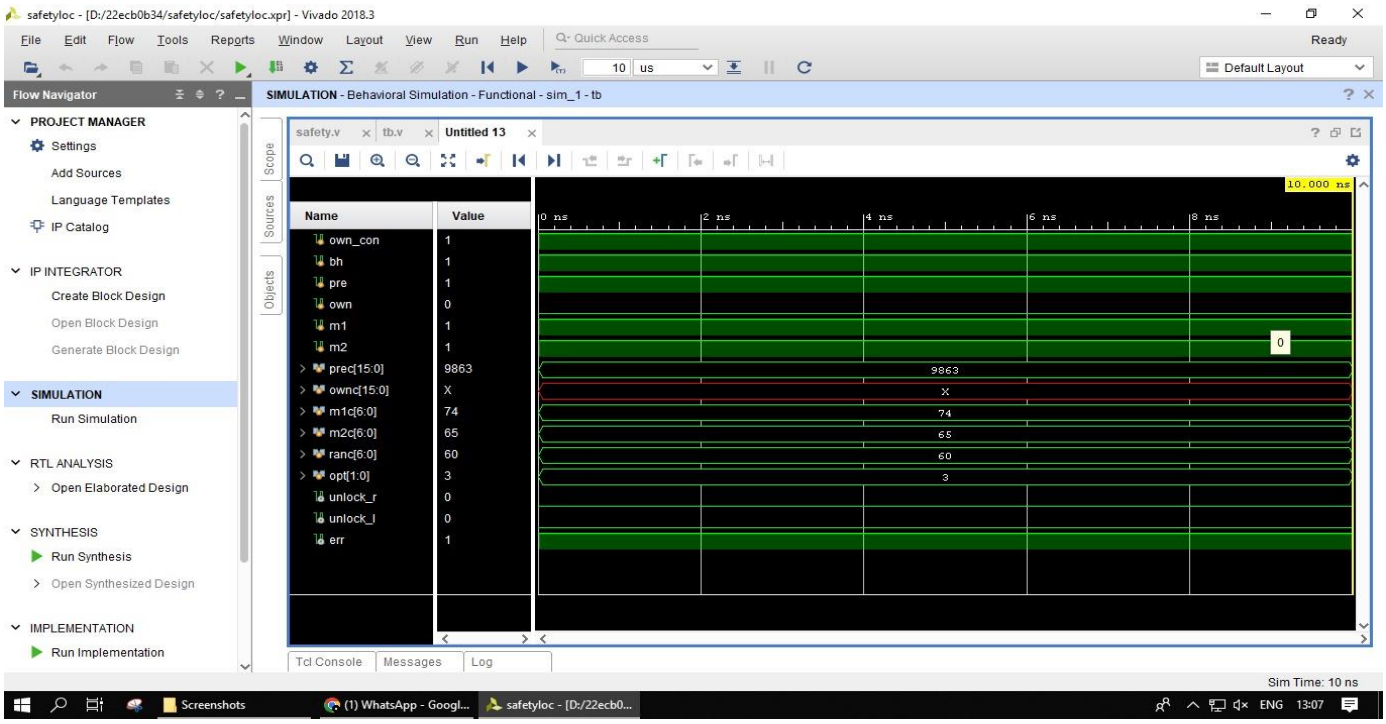
5.BH+OWNER+MANAGER1+MANAGER2(INCORRECT PASSWORD)



6.BH+OWNER+PRESIDENT(INCORRECT PASSWORD)



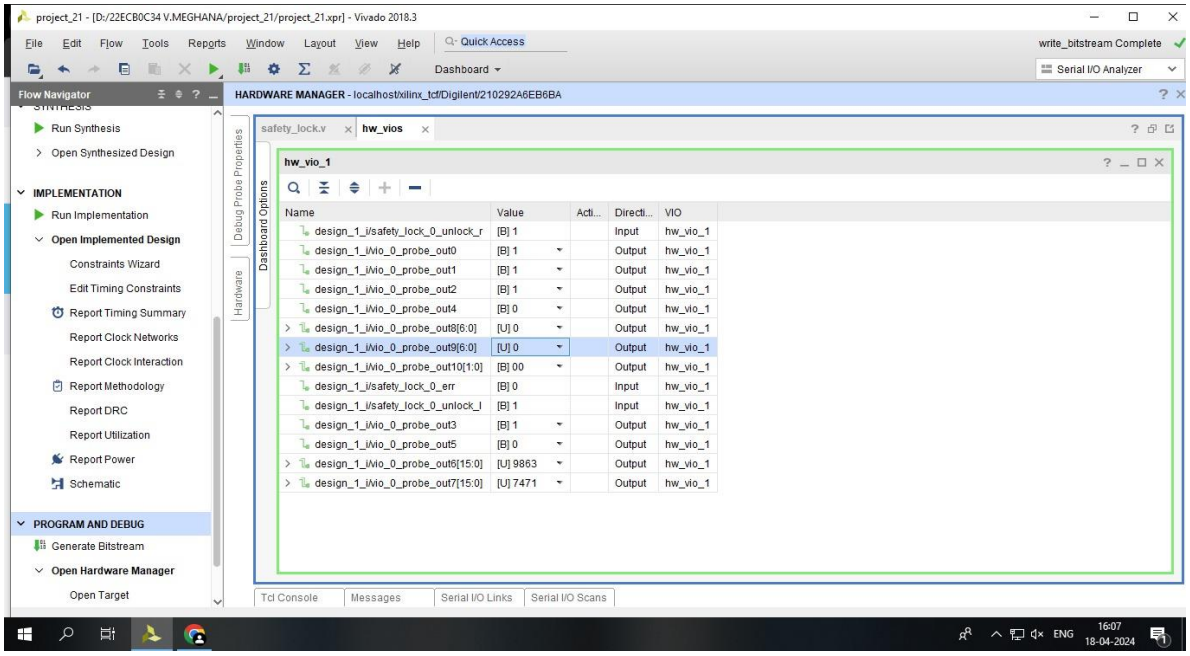
7. BH+PRESIDENT+MANAGER1+MANGER2+RANDOM GENERATOR(CORECT PASSWORD)



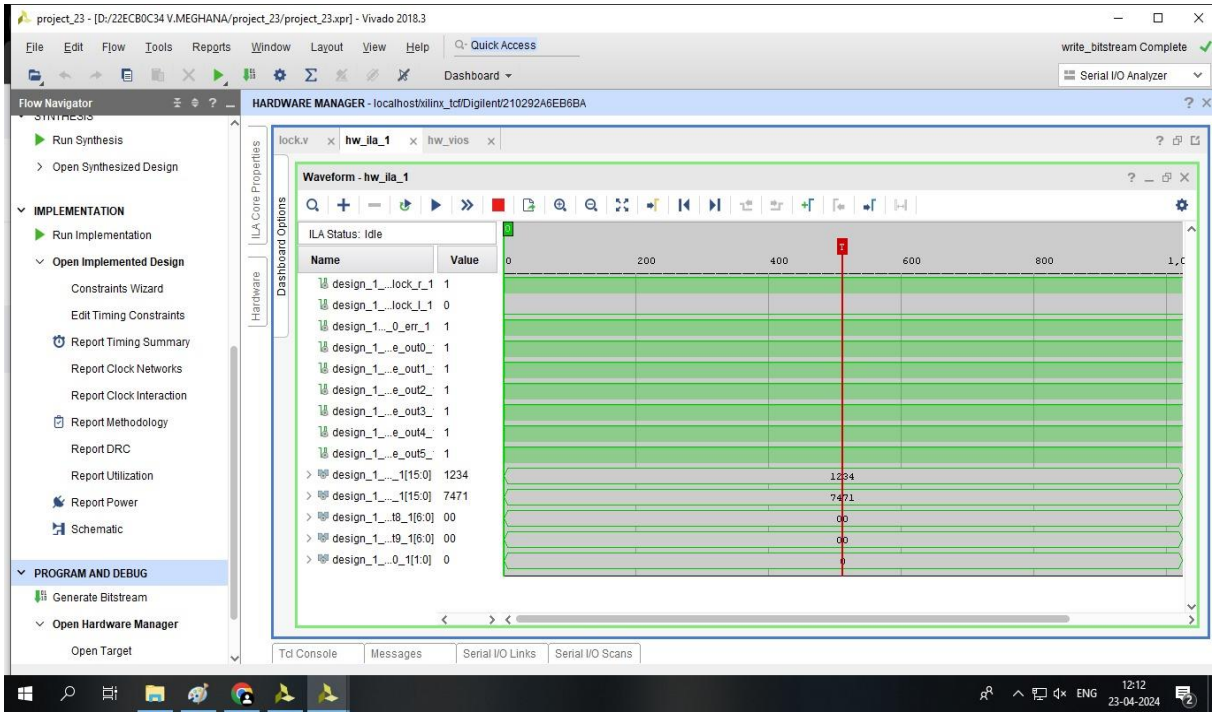
8. BH+PRESIDENT+MANAGER1+MANGER2+RANDOM GENERATOR(INCORECT PASSWORD)

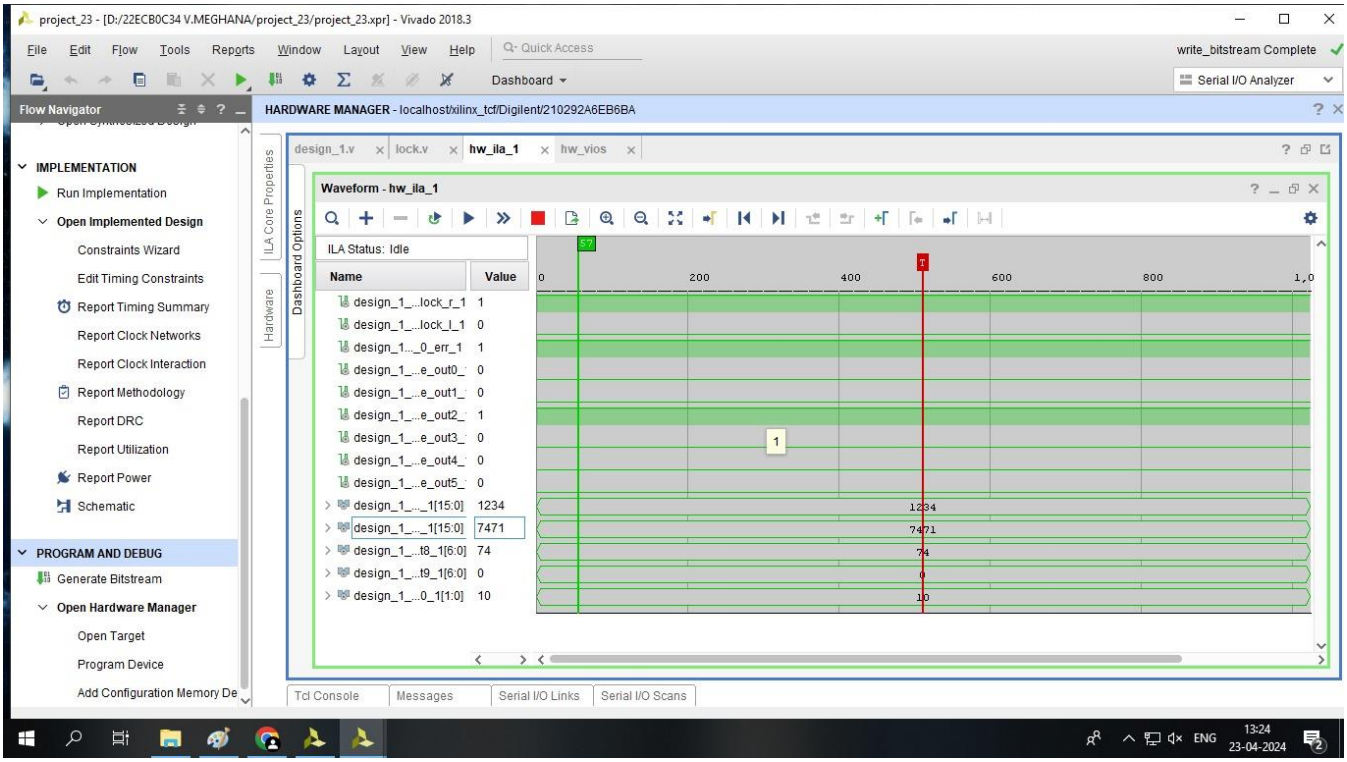
IMPLEMENTATION ON ARTIX-7 FPGA BOARD

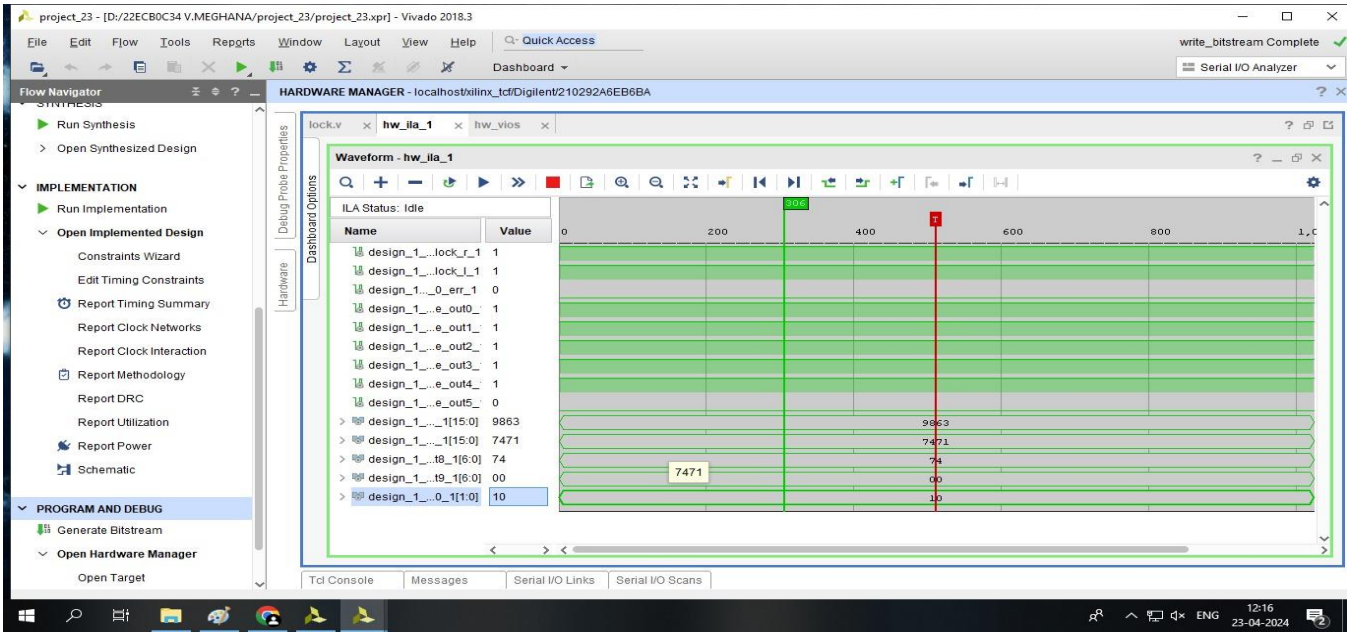
BH+PRESIDENT+OWNER(INCORRECT PASSWORD)

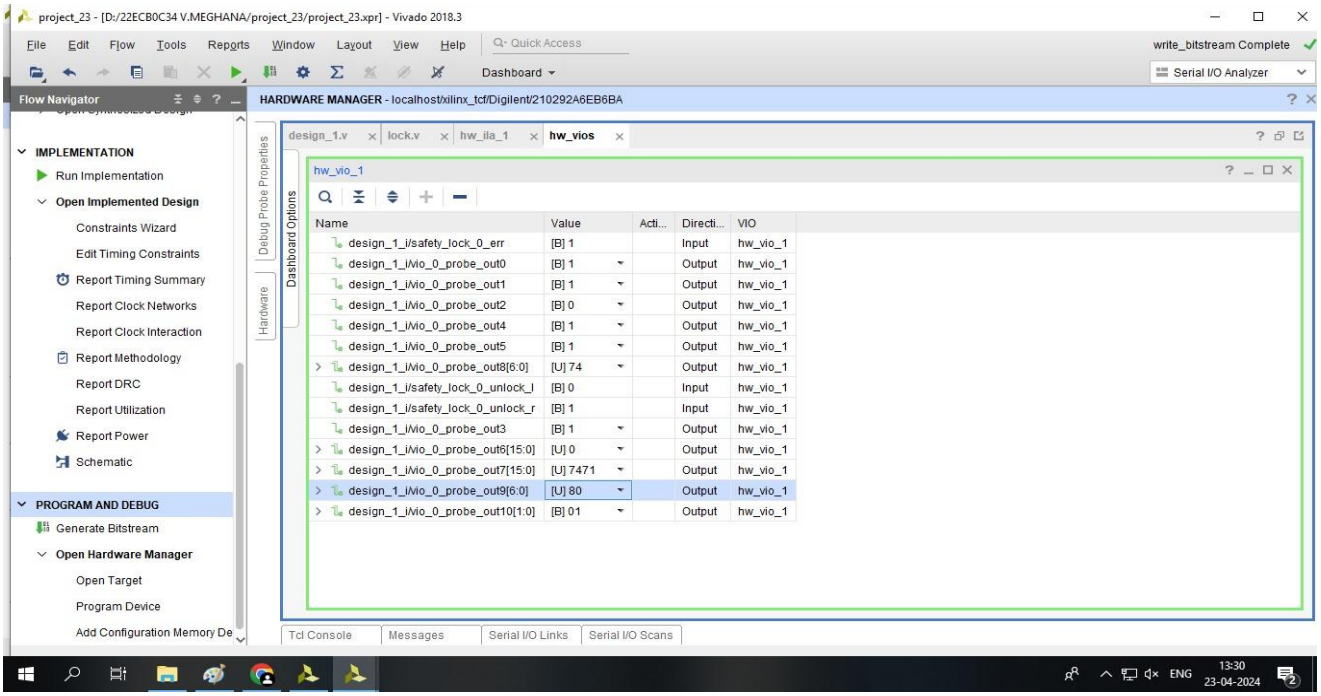


BH+OWNER+PRESIDENT (CORRECT PASSWORD)

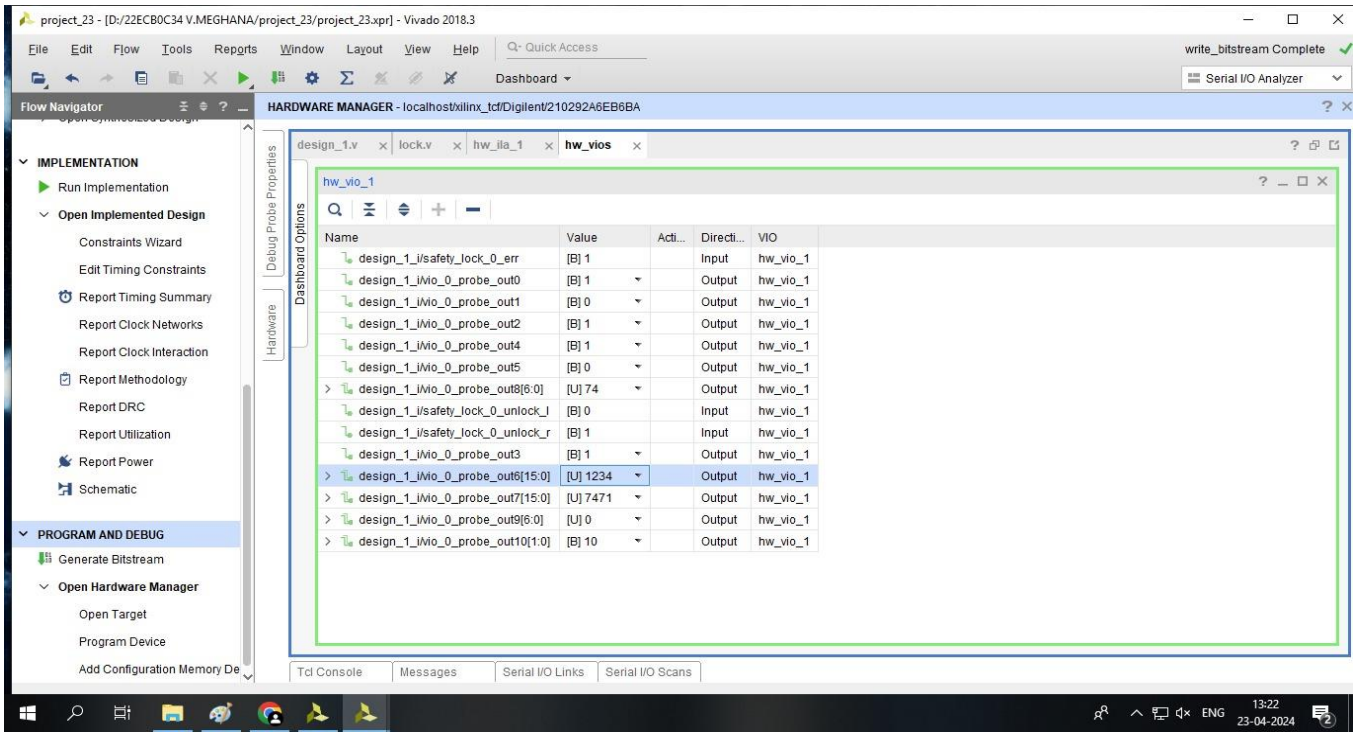

project_23 - [D:/22ECB0C34 V.MEGHANA/project_23/project_23.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
write_bitstream Complete
Dashboard
Serial I/O Analyzer
Flow Navigator
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292A6EB6BA
lock.v
hw_ila_1
hw_vios
Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_...lock_r_1 1
design_1_...lock_l_1 0
design_1_..._0_err_1 1
design_1_...e_out0_ 1
design_1_...e_out1_ 1
design_1_...e_out2_ 1
design_1_...e_out3_ 1
design_1_...e_out4_ 1
design_1_...e_out5_ 1
design_1_...__1[15:0] 1234
design_1_...__1[15:0] 7471
design_1_...t8_1[6:0] 00
design_1_...t9_1[6:0] 00
design_1_...0_1[1:0] 0
0
200
400
600
800
1234
7471
00
00
0
ILA Core Properties
Dashboard Options
Hardware
Tcl Console
Messages
Serial I/O Links
Serial I/O Scans
ENG
12:12
23-04-2024


project_23 - [D:/22ECB0C34 V.MEGHANA/project_23/project_23.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
write_bitstream Complete
Dashboard
Serial I/O Analyzer
Flow Navigator
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292A6EB6BA
IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
Program Device
Add Configuration Memory De
design_1.v
lock.v
hw_ila_1
hw_vios
Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_...lock_r_1 1
design_1_...lock_l_1 0
design_1_..._0_err_1 1
design_1_...e_out0_ 0
design_1_...e_out1_ 0
design_1_...e_out2_ 1
design_1_...e_out3_ 0
design_1_...e_out4_ 0
design_1_...e_out5_ 0
design_1_..._1[15:0] 1234
design_1_..._1[15:0] 7471
design_1_...t8_1[6:0] 74
design_1_...t9_1[6:0] 0
design_1_...0_1[1:0] 10
Tcl Console
Messages
Serial I/O Links
Serial I/O Scans
ENG
13:24
23-04-2024

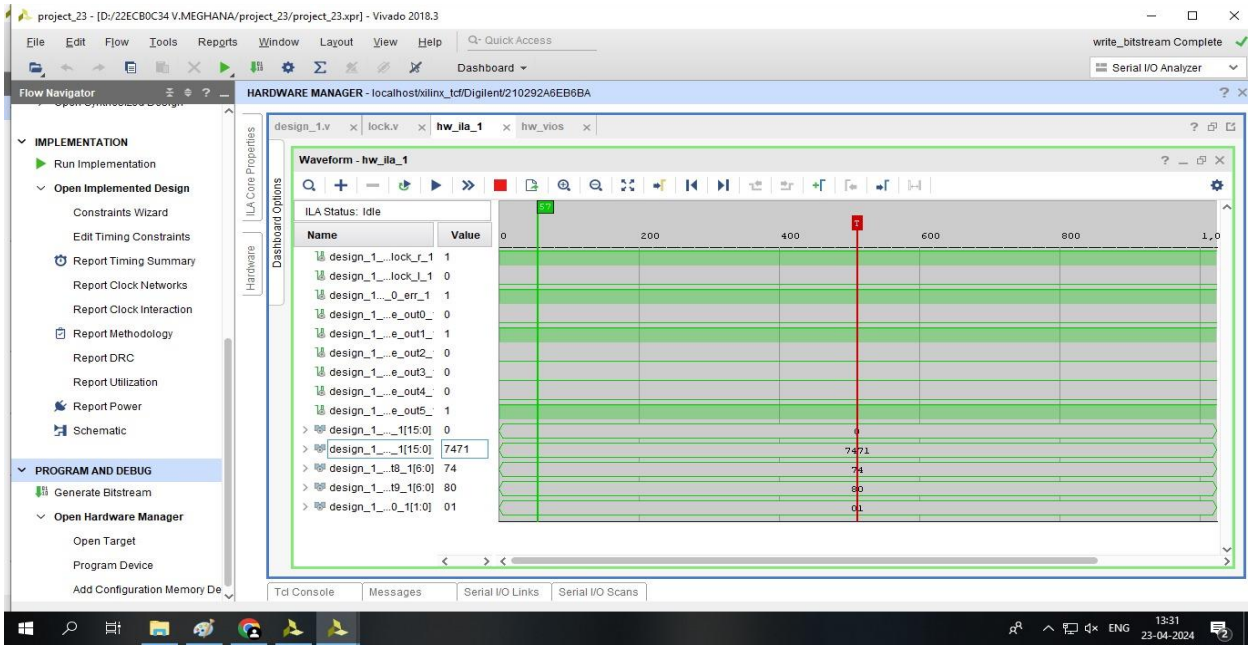

project_23 - [D:/22ECB0C34 V.MEGHANA/project_23/project_23.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
write_bitstream Complete
Dashboard
Serial I/O Analyzer
Flow Navigator
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292A6EB6BA
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
lock.v
hw_ila_1
hw_vios
Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_...lock_r_1 1
design_1_...lock_l_1 1
design_1_..._0_err_1 0
design_1_...e_out0_ 1
design_1_...e_out1_ 1
design_1_...e_out2_ 1
design_1_...e_out3_ 1
design_1_...e_out4_ 1
design_1_...e_out5_ 0
design_1_..._1[15:0] 9863
design_1_..._1[15:0] 7471
design_1_...t8_1[6:0] 74
design_1_...t9_1[6:0] 00
design_1_...0_1[1:0] 10
Tcl Console
Messages
Serial I/O Links
Serial I/O Scans
ENG
12:16
23-04-2024

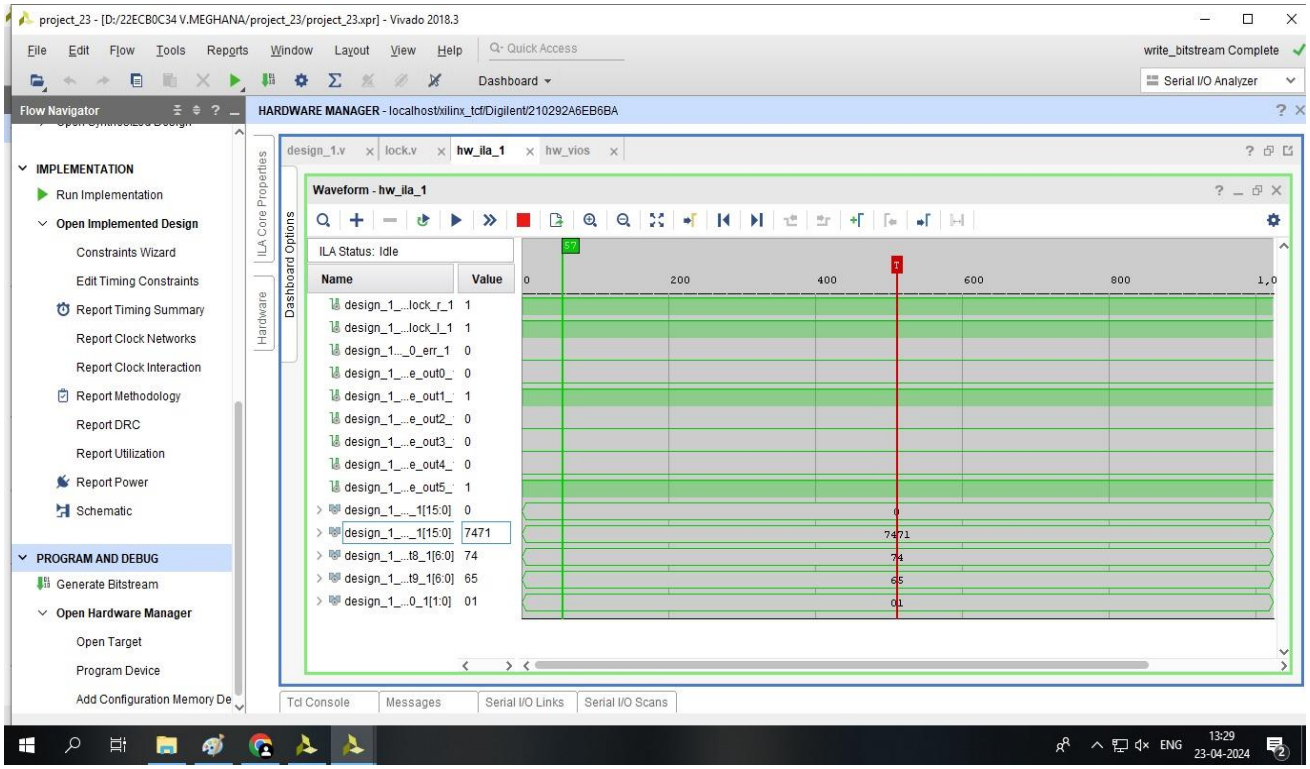


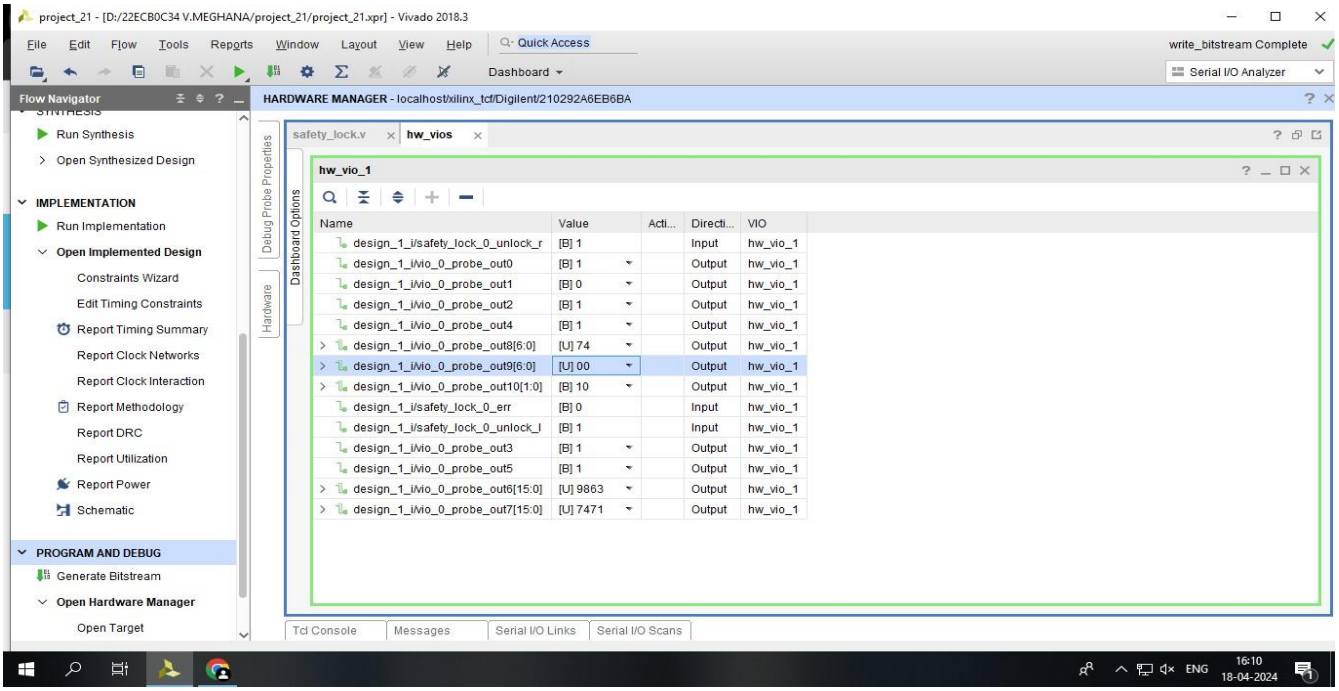
BH+OWNER+MANAGER1+MANAGER 2(INCORRECT PASSWORD)



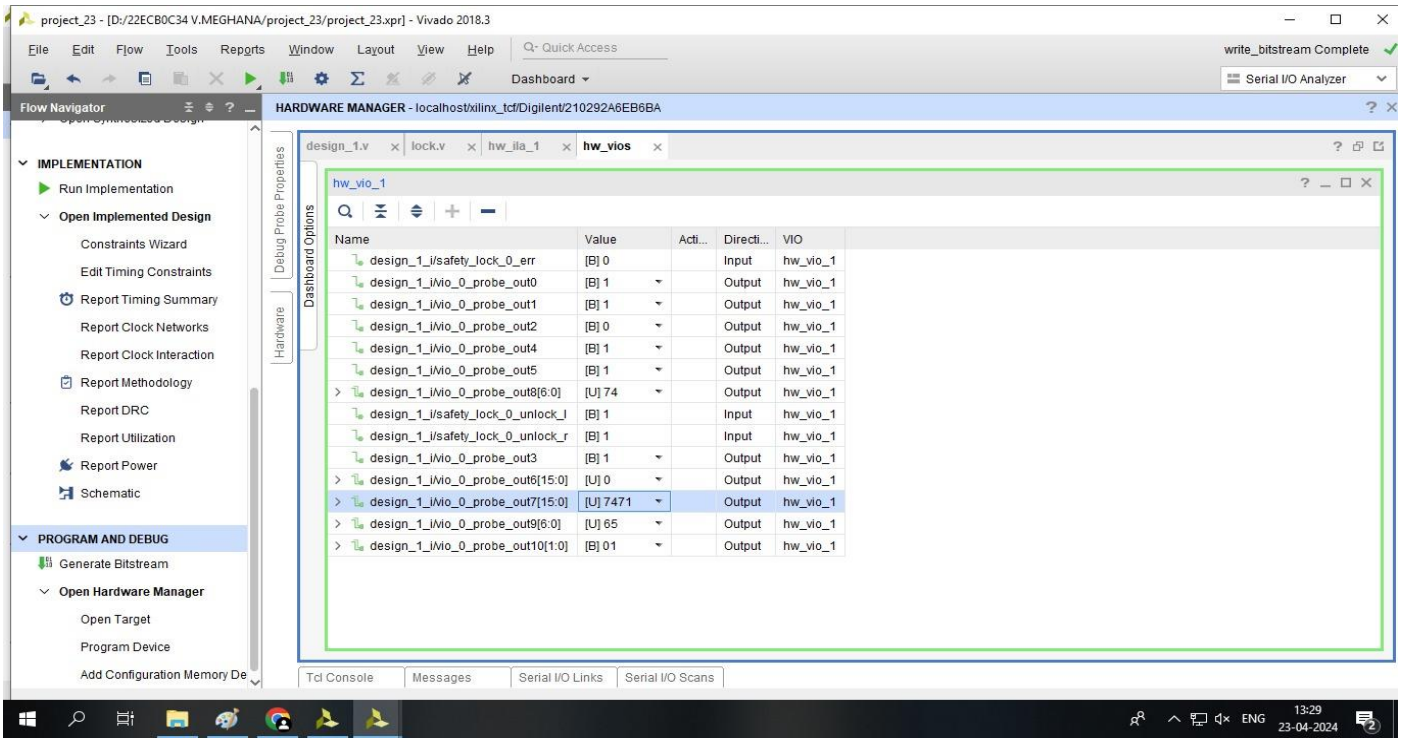
BH'+OWNER+PRESIDENT+MANAGER1(INCORRECT PASSWORD)


project_23 - [D:/22ECB0C34 V.MEGHANA/project_23/project_23.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
write_bitstream Complete
Dashboard
Serial I/O Analyzer
Flow Navigator
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292A6EB6BA
IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
Program Device
Add Configuration Memory De
design_1.v
lock.v
hw_ila_1
hw_vios
Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_...lock_r_1 1
design_1_...lock_l_1 0
design_1_..._0_err_1 1
design_1_...e_out0_ 0
design_1_...e_out1_ 1
design_1_...e_out2_ 0
design_1_...e_out3_ 0
design_1_...e_out4_ 0
design_1_...e_out5_ 1
design_1_..._1[15:0] 0
design_1_..._1[15:0] 7471
design_1_...t8_1[6:0] 74
design_1_...t9_1[6:0] 80
design_1_...0_1[1:0] 01
0
7471
74
80
01
Tcl Console
Messages
Serial I/O Links
Serial I/O Scans
ILA Core Properties
Dashboard Options
Hardware
ENG
13:31
23-04-2024


project_23 - [D:/22ECB0C34 V.MEGHANA/project_23/project_23.xpr] - Vivado 2018.3
File Edit Flow Tools Reports Window Layout View Help
Quick Access
write_bitstream Complete
Dashboard
Serial I/O Analyzer
Flow Navigator
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292A6EB6BA
IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
Program Device
Add Configuration Memory De
design_1.v
lock.v
hw_ila_1
hw_vios
Waveform - hw_ila_1
ILA Status: Idle
Name
Value
design_1_...lock_r_1 1
design_1_...lock_l_1 1
design_1_..._0_err_1 0
design_1_...e_out0_ 0
design_1_...e_out1_ 1
design_1_...e_out2_ 0
design_1_...e_out3_ 0
design_1_...e_out4_ 0
design_1_...e_out5_ 1
design_1_..._1[15:0] 0
design_1_..._1[15:0] 7471
design_1_...t8_1[6:0] 74
design_1_...t9_1[6:0] 65
design_1_...0_1[1:0] 01
0
7471
74
65
01
Tcl Console
Messages
Serial I/O Links
Serial I/O Scans
ILA Core Properties
Dashboard Options
Hardware
ENG
13:29
23-04-2024



BH'+OWNER+PRESIDENT+MANAGER1(CORRECT PASSWORD)



BH+OWNER+MANAGER1+MANGER2(CORRECT PASSWORD)

CONCLUSION

This project successfully demonstrates the implementation of a Safety Deposit Vault lock controller Using Verilog HDL. The system provides secure access to the vault based on different combinations of access codes, depending on the availability of the owner, bank authorities, and business hours. During business hours, the system requires an 8-digit code, with the owner and president's codes being used to unlock the safe. Outside of business hours, a 10-digit code is required, with the owner , president, and one manager's codes being used for access. In the special case of the owner being unavailable, the president and both managers can access the safe using an override 3-digit code in addition to their own codes.

The implementation of this system showcases the versatility and security features that can be Achieved through the use of digital circuits, providing a reliable and robust solution for safeguarding valuable assets in a safety deposit vault.

REFERENCES

https://www.academia.edu/84536964/Electronic_combination_lock_system_using_verilog_HDL