Mini-Project Design and Implementation in Logisim and HDL

SECURE VOTE: DIGITAL VOTING MACHINE

TEAM MEMBERS:

• 221CS118, GNANA JYOTHI, chinthagnanajyothi.221cs118@nitk.edu.in, 9346796854

• 221CS139, P.HASITHA, prathapachandanasaisrihasitha.221cs139@nitk.edu.in, 9030495457

• 221CS156, STHUTHI S, sthuthis.221cs156@nitk.edu.in, 9980688359

ABSTRACT:

In response to the pressing need for secure and efficient voting methods, our digital voting machine (DVM) mini project is dedicated to revolutionizing the way we conduct elections. Our mission is to design a tamper-free, user-friendly electronic voting system that addresses the shortcomings of traditional voting systems and contributes to the advancement of democratic processes. Traditional methods of verifying voter authenticity and authorization are prone to complications and human errors. To overcome these challenges, we propose a state-of-the-art password-protected voting system. This innovation simplifies voter verification, enhancing the security of the electoral process.

To build trust in the system's accuracy, we have incorporated a transparent LED indicator that offers real-time confirmation to voters. This visual cue is pivotal in boosting confidence among voters. Furthermore, our robust voter verification through the password system significantly reduces the risk of fraudulent voting, thus safeguarding the integrity of elections. Electronic voting not only ensures greater security but also streamlines the entire process, effectively reducing queues and wait times. This streamlined approach has the potential to increase voter turnout, making it a more accessible and convenient experience for all citizens.

In practical terms, our DVM is activated before voting commences. It swiftly verifies voters through the password system, with an audible beep sound indicating rejection of unauthorized voters. Voters can then cast their ballots by selecting their preferred candidate through a designated button. The system confirms the successful recording of the vote with the illumination of an LED bulb. Finally, the accumulated vote count and the election's winner are displayed on a screen or printed, ensuring complete transparency throughout the electoral process.

Our digital voting machine project is a significant leap toward creating a more reliable, efficient, and secure voting experience. It empowers citizens and strengthens the democratic foundation of our society by making elections more accessible, inclusive, and trustworthy.

BRIEF DESCRIPTION:

Creating a seamless and secure online voting process is a monumental undertaking with profound implications for the accessibility and efficiency of democratic processes. In an increasingly digital age, the convenience and ease of the internet have driven people to conduct various transactions and activities online, and voting is no exception to this trend. Transitioning to an online voting system has the potential to yield substantial long-term savings by eliminating the need for physical voting booths, paper ballots, and the hiring of personnel to manage the voting process. However, the crux of successfully implementing online voting lies in ensuring the utmost security and integrity of the entire system.

The approach outlined here, which incorporates password verification and binary conversion of voting inputs, indeed represents an innovative and potentially secure method. To comprehensively explore this approach, we must delve deeper into its various facets, taking into account its significance in safeguarding the democratic process.

Password Verification:

The implementation of a password-based system to verify the identity of the voter is a fundamental pillar of online voting security. Requiring each voter to enter a unique password before their vote is counted serves the vital purpose of confirming the identity of the individual, thus preventing unauthorized participation. Employing XNOR logic gates to compare the entered password

with the one stored in the database adds a layer of robustness to the security infrastructure. This mechanism ensures that only when all binary values output '1' (indicating a perfect match between the entered and stored password) is the vote considered valid. This robust authentication method makes it exceedingly difficult for malicious actors to compromise the system, significantly enhancing its security.

Binary Conversion and Tamper-Proof Results:

The concept of converting voting inputs into a binary format and processing them through a counter to establish the final vote tally is a novel and intriguing technique. This approach brings a degree of certainty to the process, guaranteeing that votes are accurately counted and the results remain tamper-proof. In this system, when a voter selects a candidate, the associated clock signal is set to '1', while all others are set to '0'. This mechanism allows the system to not only keep track of individual votes but also ensures that a voter can only cast a single vote. The freezing of votes after each voter has exercised their right and the subsequent display of winners maintain transparency and accountability throughout the election process.

However, it's imperative to recognize that while these methods offer innovation and the potential to bolster the security and efficiency of the voting process, the security of an online voting system can never be guaranteed with absolute certainty. The cybersecurity landscape is in a perpetual state of flux, with new vulnerabilities surfacing at any given moment. Therefore, a commitment to regularly evaluating the system and engaging with security experts is not only wise but a necessity. This ongoing vigilance and collaboration with experts serve to pinpoint and address vulnerabilities as they emerge, ensuring that the system remains robust, trustworthy, and adaptive to evolving threats.

In addition to the technical facets, the human elements of online voting security should not be overlooked. Ensuring that voters are well-informed about best practices for safeguarding their passwords and maintaining the security of their devices is of paramount importance. Strong voter education campaigns can play a pivotal role in protecting the integrity of the online voting process by empowering voters with the knowledge to uphold their end of the security bargain.

The successful implementation of such a system necessitates a resilient infrastructure capable of accommodating the anticipated load, along with well-devised contingency plans to manage unforeseen events, including system failures and cyberattacks. Moreover, the safeguarding of voter data and

anonymity is of utmost significance. The system must be designed to guarantee that individual voting choices are kept confidential, maintaining the cornerstone of a democratic process - the secret ballot.

In conclusion, the quest for a seamless and secure online voting system is an endeavor that merits commendation. By amalgamating innovative technologies such as binary conversion and password verification with continuous evaluation and expert collaboration, we are taking the steps required to ensure that the voting process remains steadfast, transparent, and resilient in the face of potential security challenges. Ultimately, this can lead to a more accessible and efficient democratic process, enhancing the trust and participation of citizens in the electoral process. The road ahead may be fraught with challenges, but with dedication, collaboration, and adaptability, the realization of a secure online voting system that upholds the integrity of democracy is indeed within reach.

WORKING:

comparator Module:

This module is designed to compare two sets of 4-bit inputs, A and B, and produce an output 'e' based on the comparison results. For each bit (0 to 3) in A and B, it calculates the XNOR of the bits. The inverted XOR results for each bit are stored in a 4-bit wire 'x'. Finally, 'e' is set to 1 if all bits in 'x' are 1; otherwise, 'e' is set to 0. That is, e is one if all the corresponding bits in A are equal to the corresponding bits of B.

password Module:

The password module combines four instances of the comparator module to compare the full four digit password. The output o of this module is the logical AND of the e outputs from the four comparator modules. This means that o will be 1 if and only if all four comparators agree on the vote. Hence, output o will be 1 if the user enters the right password.

Voting_machine module:

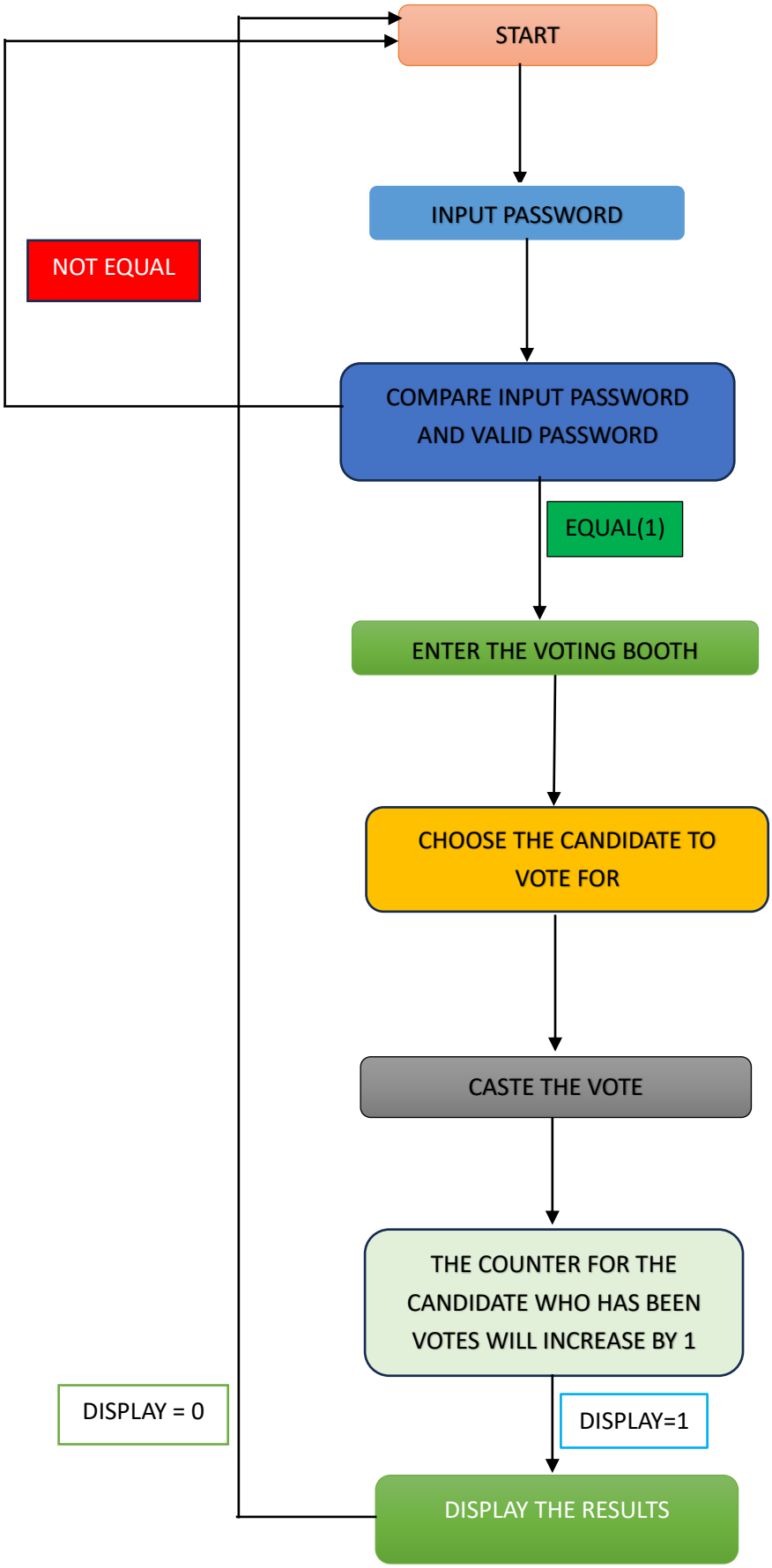
This module instantiates the password module to decide the votes based on the input password and valid password. We will be using four clock signals clk1, clk2, clk3, clk4 to ensure smooth voting process. When all the clk signals are set to 0 and display is set to zero, counters the votes for each candidates will be zero. Now if the user enters the right password , then o is 1. Now the variables count1, count2, count3, count4 store the number of votes for each candidate in previous state. When clock signal for one candidate is set to 1, rest of the clock

signals are set to 0, hence it removes the chance of multiple voting . At the end of the voting process, if the display is set 1, then the voting machine compares the totals number of votes for each candiates and displays the results.

FUNCTIONAL TABLE:

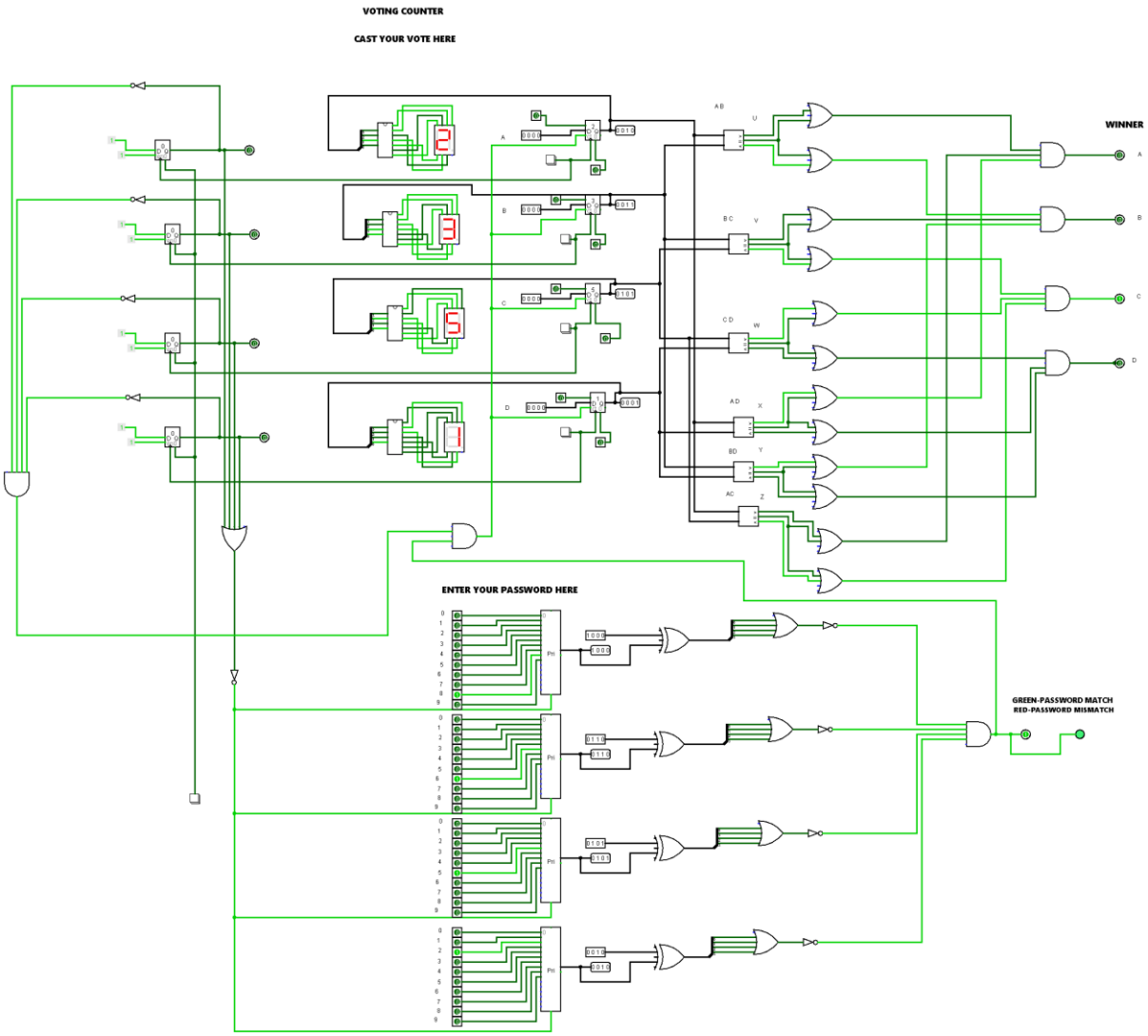
A	B	O	clk1	clk2	clk3	clk4	counter1	counter2	counter3	counter4
2987	2987	1	0	0	0	0	0	0	0	0
2987	2987	1	1	0	0	0	1	0	0	0
8421	8420	0	0	0	1	0	1	0	0	0
2345	2345	1	0	0	0	1	1	0	0	1
2987	2987	1	0	0	1	0	1	0	1	1
3217	2987	0	0	1	0	0	1	0	1	1
2987	3597	0	1	0	0	0	1	0	1	1
8421	8421	1	0	0	1	0	1	0	2	1
2345	9745	0	0	0	0	1	1	0	2	1
2987	3687	0	0	0	1	0	1	0	2	1
1122	1122	1	0	1	0	0	1	1	2	1
1122	1122	1	0	1	0	0	1	1	2	1
2987	2987	1	1	0	0	0	2	1	2	1
8421	8240	0	0	0	1	0	2	1	2	1
2345	2345	1	0	0	0	1	2	1	2	2
2987	2987	1	0	0	1	0	2	1	3	2
3217	2987	0	0	1	0	0	2	1	3	2
2987	3597	0	1	0	0	0	2	1	3	2
8421	8421	1	0	0	1	0	2	1	4	2
2345	8745	0	0	0	0	1	2	1	4	2
2987	3687	0	0	0	1	0	2	1	4	2
1122	1122	1	0	1	0	0	2	2	4	2

FLOWCHART

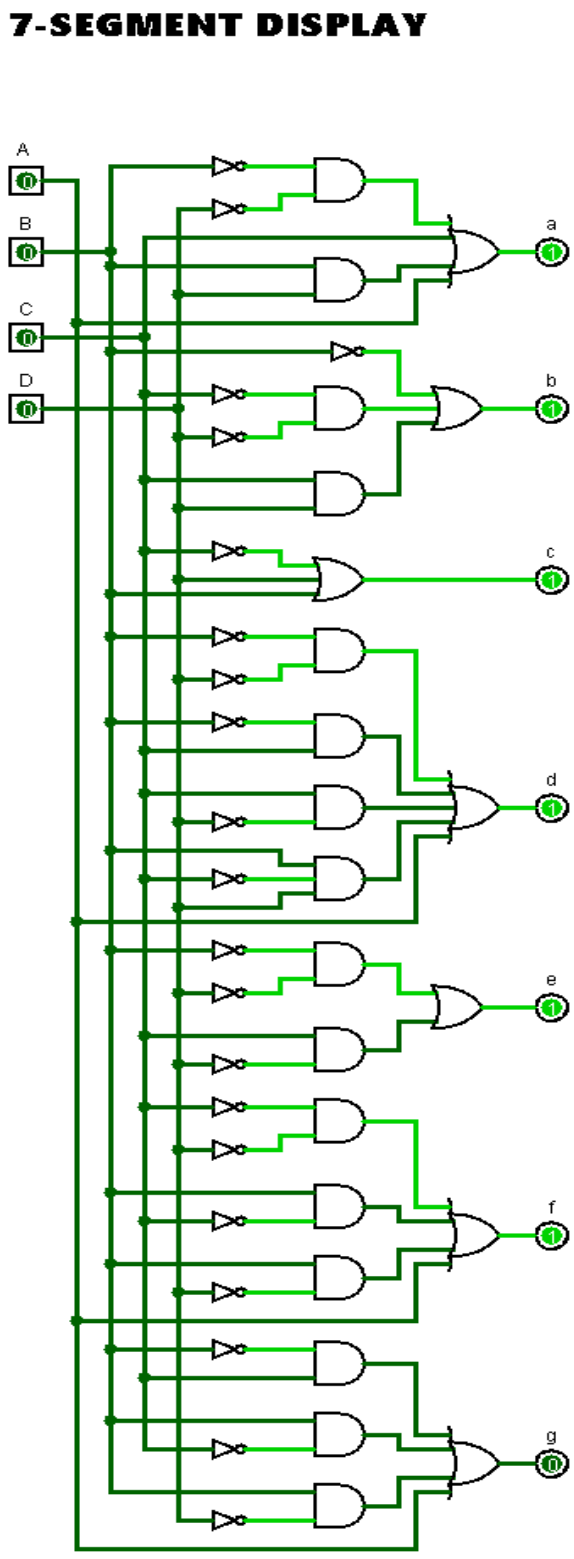

START
INPUT PASSWORD
NOT EQUAL
COMPARE INPUT PASSWORD AND VALID PASSWORD
EQUAL(1)
ENTER THE VOTING BOOTH
CHOOSE THE CANDIDATE TO VOTE FOR
CASTE THE VOTE
THE COUNTER FOR THE CANDIDATE WHO HAS BEEN VOTES WILL INCREASE BY 1
DISPLAY = 0
DISPLAY=1
DISPLAY THE RESULTS

LOGISIM CIRCUIT DIAGRAM:

VOTING MACHINE:



7-Segment Display:



VERILOG CODE:

DATAFLOW MODEL

```
module comparator(input [3:0]A,input [3:0]B,output e);

wire [3:0]x;

assign x[0]=~(A[0]^B[0]);

assign x[1]=~(A[1]^B[1]);

assign x[2]=~(A[2]^B[2]);

assign x[3]=~(A[3]^B[3]);

assign e=x[0]&x[1]&x[2]&x[3];

endmodule


module password(input [3:0]A1,input [3:0]A2,input [3:0]A3,input
[3:0]A4,input [3:0]B1,input [3:0]B2,input [3:0]B3,input [3:0]B4,output o);

wire e1,e2,e3,e4;

comparator C1(A1,B1,e1);

comparator C2(A2,B2,e2);

comparator C3(A3,B3,e3);

comparator C4(A4,B4,e4);

assign o= e1 & e2 & e3 & e4;

endmodule
```

```
module voting_machine (

    input [3:0]A1,input [3:0]A2,input [3:0]A3,input [3:0]A4,input [3:0]B1,input
[3:0]B2,input [3:0]B3,input [3:0]B4,output o,

    input wire clk1,

    input wire clk2,

    input wire clk3,

    input wire clk4,

    input wire [3:0]count1,

    input wire [3:0]count2,

    input wire [3:0]count3,

    input wire [3:0]count4,

    output reg [3:0] counter1,

    output reg [3:0] counter2,

    output reg [3:0] counter3,

    output reg [3:0] counter4,

    input wire display,

    output reg P,

    output reg Q,

    output reg R,
```

```
    output reg S
);
password p1(A1,A2,A3,A4,B1,B2,B3,B4,o);
always@(clk1==0 & clk2==0 & clk3==0 & clk4==0 &o==1'b1 & display==0)
begin
    counter1 <= 4'b0000;
    counter2 <= 4'b0000;
    counter3 <= 4'b0000;
    counter4 <= 4'b0000;
end
always@(clk1==0 & clk2==0 & clk3==0 & clk4==0 &o==1'b0 & display==0)
begin
    counter1 <= 4'b0000;
    counter2 <= 4'b0000;
    counter3 <= 4'b0000;
    counter4 <= 4'b0000;
end
always @(posedge clk1 & clk2==0 &clk3==0 &clk4==0 &o==1'b0 &
display==0 ) begin
    counter1 <= count1;
    counter2 <= count2;
```

```
    counter3 <= count3;

    counter4 <= count4;

end

always @(posedge clk2 & clk1==0 &clk3==0 &clk4==0 &o==1'b0 &
display==0) begin

    counter2 <= count2;

    counter1 <= count1;

    counter3 <= count3;

    counter4 <= count4;

end

always @(posedge clk3 &clk2==0 &clk1==0 &clk4==0 &o==1'b0 &
display==0) begin

    counter3 <= count3;

    counter1 <= count1;

    counter4 <= count4;

    counter2 <= count2;

end

always @(posedge clk4 & clk1==0 &clk3==0 &clk2==0 &o==1'b0 &
display==0) begin

    counter4 <= count4;

     counter1 <= count1;
```

```
     counter2 <= count2;

     counter3 <= count3;

end

always @(posedge clk1 & clk2==0 &clk3==0 &clk4==0 &o==1'b1 &
display==0 ) begin

    counter1 <= count1 + 4'b0001;

    counter2 <= count2;

    counter3 <= count3;

    counter4 <= count4;

end

always @(posedge clk2 & clk1==0 &clk3==0 &clk4==0 &o==1'b1 &
display==0) begin

    counter2 <= count2 + 4'b0001;

    counter1 <= count1;

    counter3 <= count3;

    counter4 <= count4;

end

always @(posedge clk3 &clk2==0 &clk1==0 &clk4==0 &o==1'b1 &
display==0) begin

    counter3 <= count3 + 4'b0001;

    counter1 <= count1;

    counter4 <= count4;
```

```
    counter2 <= count2;

end

always @(posedge clk4 & clk1==0 &clk3==0 &clk2==0 &o==1'b1 &
display==0) begin

    counter4 <= count4 + 4'b0001;

     counter1 <= count1;

     counter2 <= count2;

     counter3 <= count3;

end

always @(display==1'b1) begin

     if (counter1 >= counter2 && counter1 >= counter3 && counter1 >=
counter4)

              P <= 1'b1;

          else

              P <= 1'b0;



          if (counter2 >= counter1 && counter2 >= counter3 && counter2 >=
counter4)

              Q <= 1'b1;

          else

              Q <= 1'b0;
```

```
        if (counter3 >= counter1 && counter3 >= counter2 && counter3 >=
counter4)
            R <= 1'b1;
        else
            R <= 1'b0;


        if (counter4 >= counter1 && counter4 >= counter2 && counter4 >=
counter3)
            S <= 1'b1;
        else
            S <= 1'b0;
end
endmodule
```

TESTBENCH CODE:

```
module voting_machine_tb;

reg clk1, clk2, clk3, clk4,display;

reg [3:0] count1,A1,A2,A3,A4,B1,B2,B3,B4;

reg [3:0] count2;

reg [3:0] count3;

reg [3:0] count4;

wire o,P,Q,R,S;
```

```
wire [3:0] counter1;

wire [3:0] counter2;

wire [3:0] counter3;

wire [3:0] counter4;

// Instantiate the voting_machine module

voting_machine uut
(.A1(A1),.A2(A2),.A3(A3),.A4(A4),.B1(B1),.B2(B2),.B3(B3),.B4(B4),.o(o),

    .clk1(clk1),

    .clk2(clk2),

    .clk3(clk3),

    .clk4(clk4),

    .count1(count1),

    .count2(count2),

    .count3(count3),

    .count4(count4),

    .counter1(counter1),

    .counter2(counter2),

    .counter3(counter3),

    .counter4(counter4),

    .display(display),
```

```
    .P(P),
    .Q(Q),
    .R(R),
    .S(S)
);

initial begin
    $dumpfile("wave.vcd");
    $dumpvars(0, voting_machine_tb);
    clk1 = 1'b0;
    clk2 = 1'b0;
    clk3 = 1'b0;
    clk4 = 1'b0;
    display=1'b0;
    A1=4'd2;A2=4'd9;A3=4'd8;A4=4'd7;B1=4'd2;B2=4'd9;B3=4'd8;B4=4'd7;
    count1<=4'b0000;
    count2<=4'b0000;
    count3<=4'b0000;
    count4<=4'b0000;
    $display("------------------------------------------------------------------------
------------------------------------------------------------");
```

```
    $display("| INPUT PASSWORD | VALID PASSWORD | ELIGIBILITY | BUTTON-A |   COUNT-A   | BUTTON-B |   COUNT-B   | BUTTON-C | COUNT-C   | BUTTON-D |   COUNT-D   |");

    $display("|       A       |       B       | match(o) |    clk1 |   counter1 |   clk2   | counter2 |   clk3   |   counter3 |   clk4   |   counter4  |");

    $display("-------------------------------------------------------------------------------------------------------------------------------");

    //$monitor("%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d", $time, counter1,counter2, clk2,counter3, clk3,counter4, clk4);

    repeat (2) begin

        $monitor("|  %d %d %d %d   | %d %d %d %d   |       %b     |    %d     |    %d     |    %d     |    %d     |    %d     |    %d     |    %d     |",A1,A2,A3,A4,B1,B2,B3,B4,o, clk1,counter1,clk2,counter2, clk3, counter3, clk4,counter4);

        #10 A1=4'd2;A2=4'd9;A3=4'd8;A4=4'd7;B1=4'd2;B2=4'd9;B3=4'd8;B4=4'd7;clk1=1'b1; clk2=1'b0; clk3=1'b0; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

        #10 A1=4'd8;A2=4'd4;A3=4'd2;A4=4'd1;B1=4'd8;B2=4'd2;B3=4'd4;B4=4'd0;clk1=1'b0; clk2=1'b0; clk3=1'b1; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;
```

```
        #10 count3<=counter3;

        #10 count4<=counter4;
        #10
A1=4'd2;A2=4'd3;A3=4'd4;A4=4'd5;B1=4'd2;B2=4'd3;B3=4'd4;B4=4'd5;clk1=
1'b0; clk2=1'b0; clk3=1'b0; clk4=1'b1;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

       #10
A1=4'd2;A2=4'd9;A3=4'd8;A4=4'd7;B1=4'd2;B2=4'd9;B3=4'd8;B4=4'd7;clk1=
1'b0; clk2=1'b0; clk3=1'b1; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;

       #10 count3<=counter3;

        #10 count4<=counter4;

        #10
A1=4'd3;A2=4'd2;A3=4'd1;A4=4'd7;B1=4'd2;B2=4'd9;B3=4'd8;B4=4'd7;clk1=
1'b0; clk2=1'b1; clk3=1'b0; clk4=1'b0;

       #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;
```

```
        #10
A1=4'd2;A2=4'd9;A3=4'd8;A4=4'd7;B1=4'd3;B2=4'd5;B3=4'd9;B4=4'd7;clk1=
1'b1; clk2=1'b0; clk3=1'b0; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

        #10
A1=4'd8;A2=4'd4;A3=4'd2;A4=4'd1;B1=4'd8;B2=4'd4;B3=4'd2;B4=4'd1;clk1=
1'b0; clk2=1'b0; clk3=1'b1; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

        #10
A1=4'd2;A2=4'd3;A3=4'd4;A4=4'd5;B1=4'd8;B2=4'd7;B3=4'd4;B4=4'd5;clk1=
1'b0; clk2=1'b0; clk3=1'b0; clk4=1'b1;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

        #10
A1=4'd2;A2=4'd9;A3=4'd8;A4=4'd7;B1=4'd3;B2=4'd6;B3=4'd8;B4=4'd7;clk1=
1'b0; clk2=1'b0; clk3=1'b1; clk4=1'b0;
```

```
        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

        #10
A1=4'd1;A2=4'd1;A3=4'd2;A4=4'd2;B1=4'd1;B2=4'd1;B3=4'd2;B4=4'd2;clk1=
1'b0; clk2=1'b1; clk3=1'b0; clk4=1'b0;

        #10 count1<=counter1;

        #10 count2<=counter2;

        #10 count3<=counter3;

        #10 count4<=counter4;

    end

    #1000;

    display=1'b1;

    $display("----------------------------------------------------------------------------
------------------------------------------------------------------");

    $display("WINNER:");

    $display("A   B    C    D");

    $monitor("%b   %b    %b    %b",P,Q,R,S);

    #10 $display("----------------------------------------------------------------------------
------------------------------------------------------------------");

end
endmodule
```

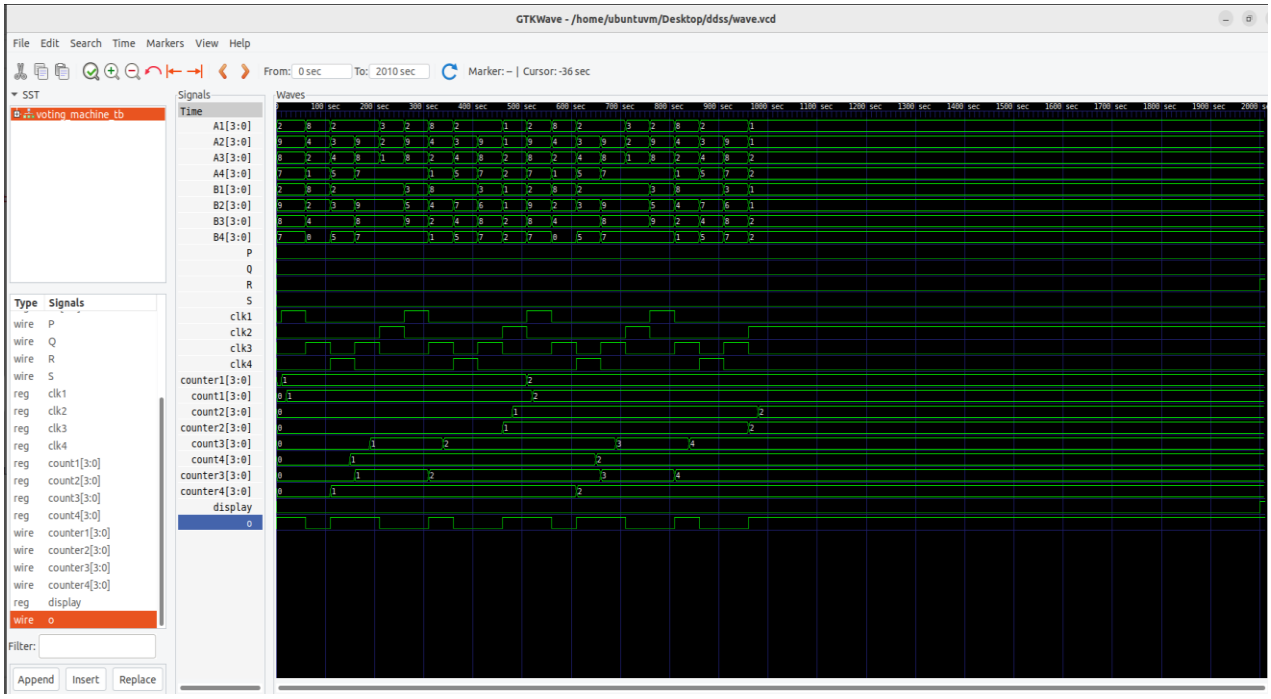
VERILOG OUTPUT:

```
ubuntuvm@UBUNTUVM:~/Desktop/ddss$ vvp output.vvp
VCD info: dumpfile wave.vcd opened for output.
```

INPUT PASSWORD A	VALID PASSWORD B	ELIGIBILITY match(o)	BUTTON-A clk1	COUNT-A counter1	BUTTON-B clk2	COUNT-B counter2	BUTTON-C clk3	COUNT-C counter3	BUTTON-D clk4	COUNT-D counter4
2 9 8 7	2 9 8 7	1	0	0	0	0	0	0	0	0
2 9 8 7	2 9 8 7	1	1	1	0	0	0	0	0	0
8 4 2 1	8 2 4 0	0	0	1	0	0	1	0	0	0
2 3 4 5	2 3 4 5	1	0	1	0	0	0	0	1	1
2 9 8 7	2 9 8 7	1	0	1	0	0	1	1	0	1
3 2 1 7	2 9 8 7	0	0	1	1	0	0	1	0	1
2 9 8 7	3 5 9 7	0	1	1	0	0	0	1	0	1
8 4 2 1	8 4 2 1	1	0	1	0	0	1	2	0	1
2 3 4 5	8 7 4 5	0	0	1	0	0	0	2	1	1
2 9 8 7	3 6 8 7	0	0	1	0	0	1	2	0	1
1 1 2 2	1 1 2 2	1	0	1	1	1	0	2	0	1
1 1 2 2	1 1 2 2	1	0	1	1	1	0	2	0	1
2 9 8 7	2 9 8 7	1	1	2	0	1	0	2	0	1
8 4 2 1	8 2 4 0	0	0	2	0	1	1	2	0	1
2 3 4 5	2 3 4 5	1	0	2	0	1	0	2	1	2
2 9 8 7	2 9 8 7	1	0	2	0	1	1	3	0	2
3 2 1 7	2 9 8 7	0	0	2	1	1	0	3	0	2
2 9 8 7	3 5 9 7	0	1	2	0	1	0	3	0	2
8 4 2 1	8 4 2 1	1	0	2	0	1	1	4	0	2
2 3 4 5	8 7 4 5	0	0	2	0	1	0	4	1	2
2 9 8 7	3 6 8 7	0	0	2	0	1	1	4	0	2
1 1 2 2	1 1 2 2	1	0	2	1	2	0	4	0	2

```
WINNER:
A   B    C    D
0   0    1    0
```

GTKWAVE:



REFERENCES:

https://nevonprojects.com/fingerprint-voting-system-project/

https://circuitdigest.com/microcontroller-projects/fingerprint-based-biometric-voting-machine-arduino

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7073090&queryText=fingerprint%20voting&newsearch=true

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7226054&queryText=fingerprint%20voting&newsearch=true

https://community.intel.com/t5/Intel-Quartus-Prime-Software/unsigned-decimal-to-binary-conversion-in-verilog/td-p/96698