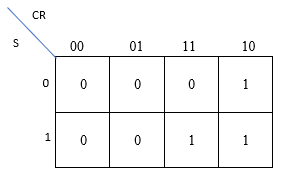
* **Objective**
* Our purpose in this experiment is study use of multiple seven segments. First, we will design its logic circuit after that we are going to write VHDL code for it and test bench code for its simulation. For the output we will display 1s and 0s on seven segment display on BASYS3.
* **Design Specification Plan**

I designed a logic circuit for a petrochemicals companies to use in their local gas stations. In every gas station there is a bank vault. Only three employees; supervisor, cashier and rookie can get inside the vault under some circumstances. The situation of getting inside of the vault will be studied. Supervisor is the only person who has the keys of the vault which means he is the only one who has the access to the vault, if s/he is in side S is 1 otherwise it is 0. Cashier is the person who makes the daily money transfer from register to bank vault and s/he is responsible from the rookie and evaluating him/her work. If cashier is in the vault C is 1 otherwise it is 0. Rookie is training to become a cashier, so s/he is observing what s/he does. If the rookie is inside the vault R is 1 otherwise it is 0. Sum of products method will be conducted, and design will be supported by Karnaugh Map and truth table.

* **Proposed Design Methodology**

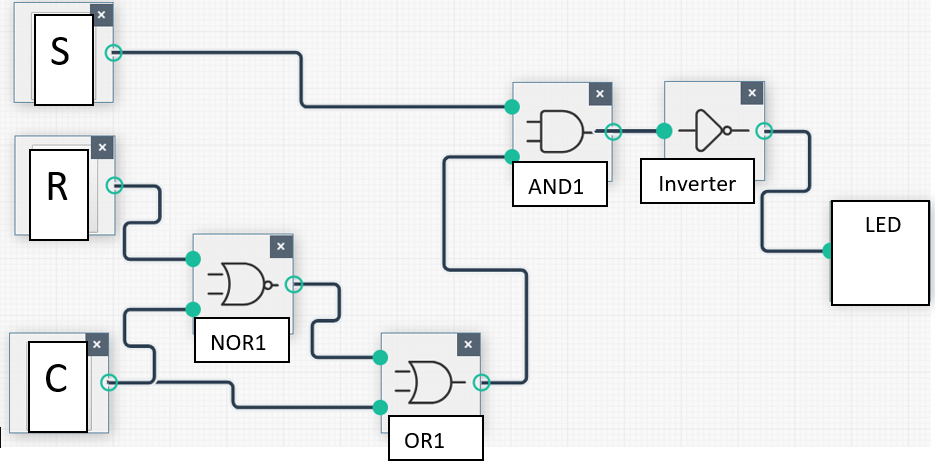
Only the supervisor has an access the bank vault so if someone else wants to get in the vault they must be together with the supervisor. Also, rookie always has to be with the cashier in order to be monitored. If rookie is in the vault cashier must be in the vault as well as the supervisor. Various letters are attained to each character in the scenario and if the entrance is allowed there will be ‘1’ displayed at the last of the seven-segment display otherwise ‘0’ and inputs will be displayed at the first 3 of the seven segment display.

K-MAP

Truth Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S | C | R | H(S,C,R) | Minterms |
| 0 | 0 | 0 | 0 |  |
| 0 | 0 | 1 | 0 |  |
| 0 | 1 | 0 | 0 |  |
| 0 | 1 | 1 | 0 |  |
| 1 | 0 | 0 | 1 |  |
| 1 | 0 | 1 | 0 |  |
| 1 | 1 | 0 | 1 |  |
| 1 | 1 | 1 | 1 |  |

Sum of products( minterms)

**Logic Schematic Diagram** 

**VHDL code:**

library IEEE; use IEEE.STD\_LOGIC\_1164.ALL;

entity wowow1 is

Port

( X : in STD\_LOGIC\_VECTOR(2 DOWNTO 0);

SSEG : out STD\_LOGIC\_VECTOR (6 DOWNTO 0);

SEG4 : out STD\_LOGIC\_VECTOR (3 DOWNTO 0));

end wowow1;

architecture Behavioral of wowow1 is

begin

process(X)

begin case X

is

when "000" => SEG4 <= "0000"; SSEG <= "1000000";

when "001" => SEG4 <= "1101"; SSEG <= "1001111";

when "010" => SEG4 <= "1011"; SSEG <= "1001111";

when "011" => SEG4 <= "1001"; SSEG <= "1001111";

when "100" => SEG4 <= "0110"; SSEG <= "1001111";

when "101" => SEG4 <= "0101";SSEG <= "1001111";

when "110" => SEG4 <= "0010";SSEG <= "1001111";

when "111" => SEG4 <= "0000"; SSEG <= "1001111";

when others=> SEG4 <= "0000"; SSEG <= "1000000";

end case;

end process;

end Behavioral;

**Test Bench Code:**

library IEEE; use IEEE.STD\_LOGIC\_1164.ALL; entity testbench is

-- Port ( );

end testbench;

architecture Behavioral of testbench is COMPONENT

wowow1 Port

( X : in STD\_LOGIC\_VECTOR(2 DOWNTO 0);

SSEG : out STD\_LOGIC\_VECTOR (6 DOWNTO 0);

SEG4 : out STD\_LOGIC\_VECTOR (3 DOWNTO 0) );

END COMPONENT;

SIGNAL X: STD\_LOGIC\_VECTOR(2 DOWNTO 0);

SIGNAL SSEG : STD\_LOGIC\_VECTOR(6 DOWNTO 0);

SIGNAL SEG4 : STD\_LOGIC\_VECTOR (3 DOWNTO 0);

BEGIN UUT: wowow1 PORT MAP

( X=>X,

SSEG=>SSEG,

SEG4=>SEG4 );

testbench: PROCESS BEGIN

wait for 100 ns;

X(0)<='1';

X(1)<='0';

X(2)<='0';

wait for 100 ns;

X(0)<='1';

X(1)<='0';

X(2)<='0';

wait for 100 ns;

X(0)<='0';

X(1)<='1';

X(2)<='1';

wait for 100 ns;

X(0)<='1';

X(1)<='1';

X(2)<='0';

wait for 100 ns;

X(0)<='0';

X(1)<='0';

X(2)<='1';

wait for 100 ns;

X(0)<='1';

X(1)<='1';

X(2)<='1';

END PROCESS;

END Behavioral;