طراحی را بر اساس فان نیومان انجام میدهیم و برای سادگی در طراحی، رجیسترهای اصلی را به صورت AX, BX, CX, DX در نظر میگیریم. با تنظیم کردن Sel میتوانیم رجیستر مورد نظر را انتخاب کنیم و با RW انتخاب کنیم این رجیستر میخواهد به روی باس بنویسد یا از آن بخواند.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;


entity ComBus is
    Port ( Sel : in  STD_LOGIC_VECTOR (2 downto 0);
           RW : in  STD_LOGIC;
           ALU: inout STD_LOGIC_VECTOR (7 downto 0);
           AX : inout  STD_LOGIC_VECTOR (7 downto 0);
           BX : inout  STD_LOGIC_VECTOR (7 downto 0);
           CX : inout  STD_LOGIC_VECTOR (7 downto 0);
           DX : inout  STD_LOGIC_VECTOR (7 downto 0);
           RAM : inout  STD_LOGIC_VECTOR (7 downto 0);
           BUS_DATA : inout  STD_LOGIC_VECTOR (7 downto 0));
end ComBus;

architecture Behavioral of ComBus is
signal R: STD_LOGIC_VECTOR (7 downto 0);
signal W: STD_LOGIC_VECTOR (7 downto 0) := "00000010";
begin

    with Sel select
        R <=    ALU when "000",
                AX  when "001",
                BX  when "010",
                CX  when "011",
                DX  when "100",
                RAM when "101",
                "ZZZZZZZZ" when others;

    with RW select
        BUS_DATA <= R when '1',
                    "ZZZZZZZZ" when others;

    with RW select
        W <=    BUS_DATA when '0',
                "ZZZZZZZZ" when others;
```

```
    ALU <=  W when (Sel = "000") else "ZZZZZZZZ";
    AX  <=  W when (Sel = "001") else "ZZZZZZZZ";
    BX  <=  W when (Sel = "010") else "ZZZZZZZZ";
    CX  <=  W when (Sel = "011") else "ZZZZZZZZ";
    DX  <=  W when (Sel = "100") else "ZZZZZZZZ";
    RAM <=  W when (Sel = "101") else "ZZZZZZZZ";

end Behavioral;
```

کد های تست را نیز به صورت زیر مینویسیم، و به این دلیل که ورودی و خروجی های ما رجیسترهای واقعی نیستند، دیتای روی باس قابل ذخیره کردن نمیباشد.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

ENTITY ComBus_test IS
END ComBus_test;

ARCHITECTURE behavior OF ComBus_test IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT ComBus
    PORT(
         Sel : IN  std_logic_vector(2 downto 0);
         RW : IN  std_logic;
         ALU : INOUT  std_logic_vector(7 downto 0);
         AX : INOUT  std_logic_vector(7 downto 0);
         BX : INOUT  std_logic_vector(7 downto 0);
         CX : INOUT  std_logic_vector(7 downto 0);
         DX : INOUT  std_logic_vector(7 downto 0);
         RAM : INOUT  std_logic_vector(7 downto 0);
         BUS_DATA : INOUT  std_logic_vector(7 downto 0)
        );
    END COMPONENT;


   --Inputs
```

```
   signal Sel : std_logic_vector(2 downto 0) := (others => '0');
   signal RW : std_logic := '0';

   --BiDirs
   signal AX : std_logic_vector(7 downto 0);
   signal BX : std_logic_vector(7 downto 0);
   signal CX : std_logic_vector(7 downto 0);
   signal DX : std_logic_vector(7 downto 0);
   signal RAM : std_logic_vector(7 downto 0);
   signal BUS_DATA : std_logic_vector(7 downto 0);

   --Outputs
   signal ALU : std_logic_vector(7 downto 0);
   -- No clocks detected in port list. Replace <clock> below with
   -- appropriate port name

BEGIN

   -- Instantiate the Unit Under Test (UUT)
   uut: ComBus PORT MAP (
          Sel => Sel,
          RW => RW,
          ALU => ALU,
          AX => AX,
          BX => BX,
          CX => CX,
          DX => DX,
          RAM => RAM,
          BUS_DATA => BUS_DATA
        );


   -- Stimulus process
   stim_proc: process
   begin
      AX <= "00000000";
      BX <= "00000001";
      CX <= "00000010";
      DX <= "00000011";

      RW <= '1';

      Sel <= "001";
      wait for 10 ns;
```

```
      Sel <= "010";
      wait for 10 ns;

      Sel <= "011";
      wait for 10 ns;

      Sel <= "100";
      wait for 10 ns;

      wait;
   end process;

END;
```

خروجی ما به صورت زیر میشود