```vhdl

---------------------Mantissa adder - Subtractor(Performs the Add/Subtract operation on Significants)--------

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

-- Sign Magnitude implementation
entity MantissaSubtractor is
    Port ( Ma : in  UNSIGNED (9 downto 0) := "0000000000"; --No of STD_LOGICs are : 4(Mantissa) + 1(Implied STD_LOGIC), sign STD_LOGIC is considered in sum_ans variable
           Mb : in  UNSIGNED (9 downto 0) := "0000000000"; --Earlier was 3 downto 0
           AddSub : in STD_LOGIC;
           Magnitude : out  UNSIGNED (9 downto 0) := "0000000000";
           Carry : out STD_LOGIC := '0' ; -- This can be generated in case of addition of 1.Ma + 1.Mb
           Sign : out STD_LOGIC);
end MantissaSubtractor;

architecture Behavioral of MantissaSubtractor is

begin
   process(AddSub,Ma,Mb)
   variable sum_ans : UNSIGNED (10 downto 0) := "00000000000";
   variable carry_ans : UNSIGNED (11 downto 0) ; -- Earlier one less than this size
   variable Manew : UNSIGNED (10 downto 0) := "00000000000"; --1 STD_LOGIC extra for knowing the sign
   variable Mbnew : UNSIGNED (10 downto 0) := "00000000000";
   variable TempOperand : UNSIGNED (10 downto 0) := "00000000000";
   variable TempCarry : UNSIGNED (11 downto 0);
   variable CarryTemp : STD_LOGIC;
   variable TempSum : UNSIGNED (9 downto 0);
   variable SignTemp : STD_LOGIC;

   begin
      carry_ans(0) := AddSub;
      Manew(9 downto 0) := Ma; --remaining STD_LOGICs as usual

      if AddSub = '1' then
         Mbnew(9 downto 0) := Mb xor "1111111111"; --2's complement of Mb (Refer the rule of subtraction using 2's complement)
      else
         Mbnew(9 downto 0) := Mb;
      end if;

      for i in 0 to 10 loop
         if (AddSub = '1' and i = 0) then
         sum_ans(i) := (Manew(i) xor Mbnew(i) xor '1');
         carry_ans(i+1) := (Manew(i) and Mbnew(i)) or (Manew(i) and '1') or (Mbnew(i) and '1');
         else
         sum_ans(i) := (Manew(i) xor Mbnew(i)) xor carry_ans(i);
         carry_ans(i+1) := (Manew(i) and Mbnew(i)) or (Manew(i) and carry_ans(i)) or (Mbnew(i) and carry_ans(i));
         end if;
      end loop;

      if AddSub = '1' then --Apply 2,s complement rules
         if sum_ans(10) = '1' then -- Carry is there, so ans is +ve
            TempSum := sum_ans(9 downto 0); --Ans is +ve, so no change
         else -- Carry is not there so ans is 2'complement of sum_ans
            sum_ans(9 downto 0) := sum_ans(9 downto 0) xor "1111111111"; -- Ans is -ve, so ans is 2's complement of itself


            for i in 0 to 10 loop
               if i = 0 then
               TempCarry(i+1) := (sum_ans(i) and TempOperand(i)) or (sum_ans(i) and '1') or (TempOperand(i) and '1');
               sum_ans(i) := (sum_ans(i) xor TempOperand(i)) xor '1';
               else
               TempCarry(i+1) := (sum_ans(i) and TempOperand(i)) or (sum_ans(i) and TempCarry(i)) or (TempOperand(i) and TempCarry(i));
               sum_ans(i) := (sum_ans(i) xor TempOperand(i)) xor TempCarry(i);
               end if;

            end loop; -- This part is to add 1 to sum_ans in order to complete the 2's complement

            --sum_ans(0) := sum_ans(0) xor '1';
            TempSum := sum_ans(9 downto 0);
         end if;
         SignTemp := not sum_ans(10); -- Zero is +ve, One is -ve, But in program it comes opposite
         CarryTemp := '0';
      else  -- As usual
         TempSum := sum_ans(9 downto 0); --Ans is +ve, so no change
         SignTemp := '0';
         if Carry_ans(11) = '1' then
         CarryTemp := Carry_ans(11); -- This is required carry for 1.Ma + 1.Mb
         else
```

```vhdl
         CarryTemp := '0';
         end if;
      end if;
      Carry <= CarryTemp;
      Magnitude <= TempSum;
      Sign <= SignTemp;

 end process;
end Behavioral;


```