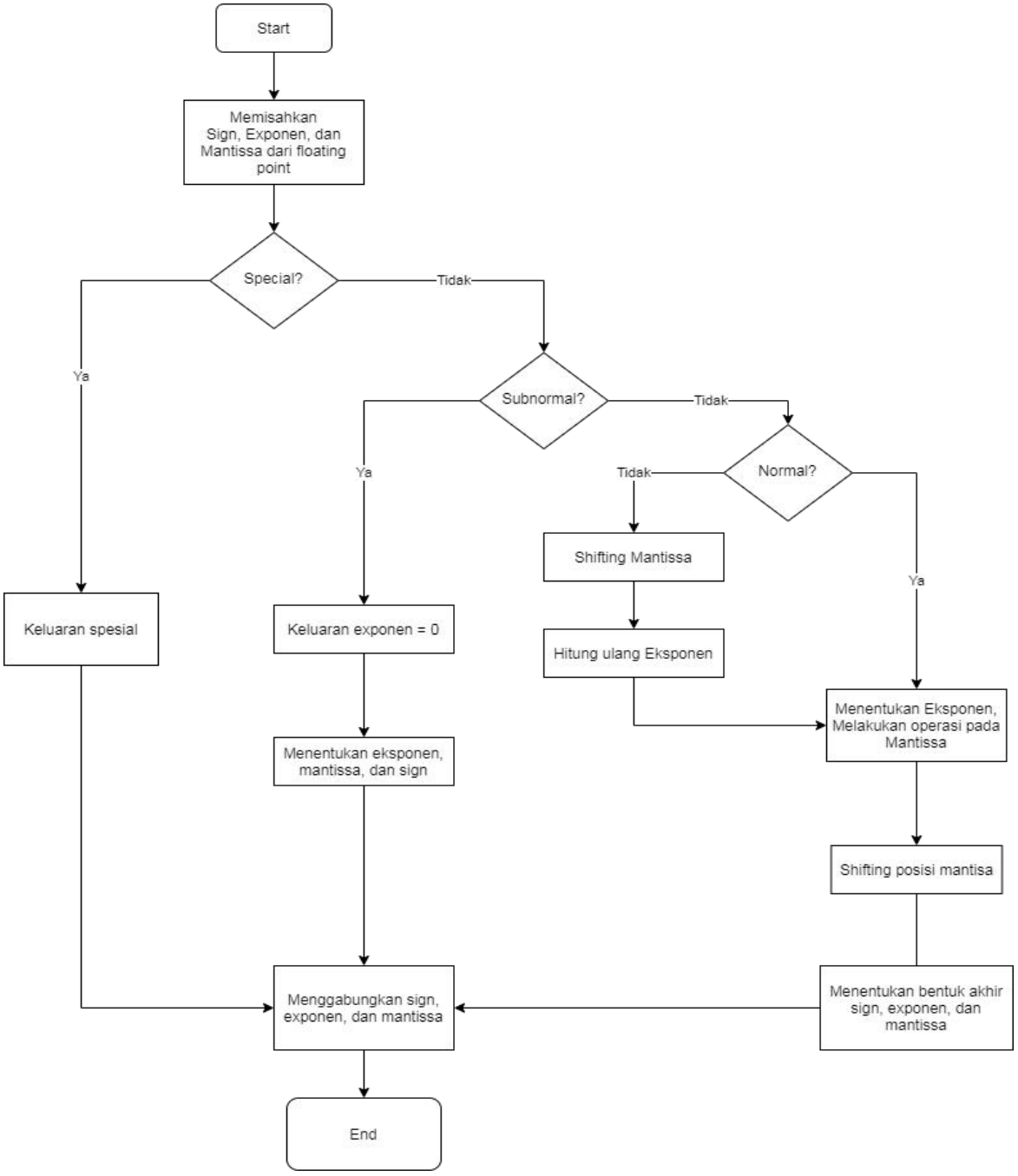
Nama : Sidartha Prastya. P

NIM :13219033

Tugas Praktikum 3

EL2102 - Sistem Digital

1. Pada praktikum ini ditugaskan untuk membuat penjumlah kombinasional half precision floating point sesuai dengan standar IEEE 754. Desain *(flowchart)* dari penjumlah ini dapat dilihat sebagai berikut.



Pola berpikir yang terdapat pada diagram alur tersebut adalah sebagai berikut.

- Pertama, floating point binary dipisahkan menjadi beberapa bagian, yaitu
    a. Sign : polaritas dari floating point (bit 16).
    b. Exponent : faktor pangkat pada floating point (bit 15 - 11).
    c. Mantissa : angka-angka penting pada floating point (bit 10 - 1).
- Setelah mengelompokkan, program melihat apakah operasi floating point tersebut masuk ke dalam kasus spesial, yaitu:
    a. Zero (nol) : sign, exponent, dan mantissa adalah nol.
    b. Infinite : mantissa berjumlah 0 dan exponent berjumlah 31.
    c. NaN : exponent berjumlah 31 dan mantissa memiliki nilai.
- Bila kasus spesial, maka program akan langsung mengeluarkan keluaran pasti, yaitu floating point itu sendiri untuk kasus Zero, infinite itu sendiri untuk kasus infinite, dan "1111110000000001" untuk kasus NaN.
- Bila bukan termasuk kasus spesial, maka kita melihat apakah floating point tersebut termasuk subnormal atau normal, yaitu:
    a. Subnormal : exponent bernilai 0 dan memiliki mantissa.
    b. Normal : exponent di antara 0 dan 31 serta memiliki mantissa.
- Bila kasus tersebut subnormal untuk kedua input, exponent output dibuat menjadi 0 dan tinggal menghitung mantissa serta menentukan sign output. Namun, apabila masuk ke dalam kasus normal, maka perlu untuk membandingkan exponent lalu melakukan *shifting* pada mantissa sesuai dengan besar perbedaan kedua exponent input.
- Setelah melakukan shifting, lakukan operasi penjumlahan manitssa dengan adder. Kemudian, tentukan sign akhir apakah positif atau negatif.
- Semua nilai akhir yang didapat (sign, exponent, dan mantissa) digabungkan menjadi output akhir.

2. Berikut adalah desain VHDL dari penjumlah tersebut. Untuk VHDL code **Fulladder, Adder-nbit,** dan **Adder-substractor**, saya menggunakan program yang sama dengan praktikum 2 yang lalu.

```
-- Nama  : Sidartha Prastya. P
-- Nim   : 13219033
-- Praktikum 3 Sistem Digital

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity Half_precision_adder is
   port (
      x_input: in std_logic_vector(15 downto 0);
      y_input: in std_logic_vector(15 downto 0);

      output   : out std_logic_vector(15 downto 0)
      );
end entity;

architecture structural of Half_precision_adder is

   component adder_substractor is
      Generic (N : integer := 10);
      port(
      OP       : in std_logic;
      x_sub    : in std_logic_vector(N-1 downto 0);
      y_sub    : in std_logic_vector(N-1 downto 0);
      result_sub  : out std_logic_vector(N-1 downto 0);
      cout_sub : out std_logic
      );
   end component;
```

Di sini saya menggambarkan Half precision adder yang tentunya memiliki 16 bit terdiri dari dua input (x dan y) serta menghasilkan sebuah output. Saya memanggil komponen adder substractor untuk melakukan perhitungan pada mantissa.

```
   signal x_man   : std_logic_vector(9 downto 0); -- Mantissa
   signal y_man   : std_logic_vector(9 downto 0);
   signal x_exp   : std_logic_vector(4 downto 0); -- Exponent
   signal y_exp   : std_logic_vector(4 downto 0);
   signal x_sign  : std_logic;                    -- Sign
   signal y_sign  : std_logic;

   signal out_x, out_y  : std_logic_vector(2 downto 0); -- Analisis Spesial
   signal SS    : std_logic;                            -- Keluaran akhir
   signal ES    : std_logic_vector(4 downto 0);
   signal MS    : std_logic_vector(9 downto 0);

   signal num_x_exp, num_y_exp : integer;               -- Besar exponen dalam integer
   signal dif_exp : integer;
   signal x_man_up : std_logic_vector(9 downto 0);      -- Mantissa hasil shifting
   signal y_man_up : std_logic_vector(9 downto 0);

   signal operator   : std_logic;                       -- Operator untuk perhitungan mantissa
   signal x_op       : std_logic_vector(9 downto 0);    -- penempatan urutan mantissa operasi hitung
   signal y_op       : std_logic_vector(9 downto 0);

   signal norm: std_logic;                              -- Status apakah normal/subnormal atau tidak
   signal cout: std_logic;

   signal MS2 : std_logic_vector(9 downto 0);           -- Mantissa akhir/keluaran khusus normal/subnormal
```

Lalu, ada beberapa signal yang dibutuhkan pada program ini, yaitu sign, exponent, dan sign untuk masing-masing input; SS, ES, MS untuk output akhir; norm, out_x dan out_y untuk mendeteksi kasus spesial; x_man_up, y_man_up untuk mantissa hasil shifting; operator, x_op, y_op, dan MS2 untuk operasi hitung floating point subnormal/normal.

```
   begin
      x_sign <= x_input(15);
      y_sign <= y_input(15);

      x_exp <= x_input(14 downto 10);
      y_exp <= y_input(14 downto 10);

      x_man <= x_input(9 downto 0);
      y_man <= y_input(9 downto 0);
```

Setelah itu, dilakukan pemisahan floating number ke dalam sign, exponent, dan mantissa.

```
out_x    <= "000" when x_exp = "00000" and x_man = 0 else                          -- x zero
            "001" when x_exp = "00000" and x_man > 0 else                          -- x subnormal
            "011" when (x_exp > "00000" and x_exp < "11111") and x_man > 0 else    -- x normal
            "100" when x_exp = "11111" and x_man = 0 else                          -- x infinite
            "110" when x_exp = "11111" and x_man > 0 else                          -- x NaN
            "000";

out_y    <= "000" when y_exp = "00000" and y_man = 0 else                          -- y zero
            "001" when y_exp = "00000" and y_man > 0 else                          -- y subnormal
            "011" when (y_exp > "00000" and y_exp < "11111") and y_man > 0 else    -- y normal
            "100" when y_exp = "11111" and y_man = 0 else                          -- y infinite
            "110" when y_exp = "11111" and y_man > 0 else                          -- y NaN
            "000";
```

Lalu dilakukan inisiasi sifat/kasus spesial.

```
process (x_sign, y_sign, out_x, out_y, dif_exp, x_man, y_man)
   begin
   -------------------------------- Zero
   if (out_x = "000") then
      SS <= y_sign;
      ES <= y_exp;
      MS <= y_man;
      norm <= '0';


   elsif (out_y = "000") then
      SS <= x_sign;
      ES <= x_exp;
      MS <= x_man;
      norm <= '0';

   end if;
   -------------------------------- Infinite
   if (out_x(0) = '1' and out_y = "100") then
      SS <= y_sign;
      ES <= y_exp;
      MS <= y_man;
      norm <= '0';

   elsif (out_y(0) = '1' and out_x = "100") then
      SS <= x_sign;
      ES <= x_exp;
      MS <= x_man;
      norm <= '0';


   end if;

   if ((out_x and out_y) = "100" and x_sign = y_sign) then
      SS <= x_sign;
      ES <= x_exp;
      MS <= x_man;
      norm <= '0';


   -------------------------------- NaN

   elsif ((out_x and out_y) = "100" and x_sign /= y_sign) then
      SS <= '1';
      ES <= "11111";
      MS <= "0000000001";
      norm <= '0';

   end if;

   if (out_x = "110" or out_y = "110") then
      SS <= '1';
      ES <= "11111";
      MS <= "0000000001";
      norm <= '0';
   end if;
```

Dilakukan klasifikasi kasus spesial beserta keluarannya.

```
    -------------------------------- Normal/Subnormal
    if ((out_x(0) and out_y(0)) = '1') then
       norm <= '1';
       num_x_exp <= conv_integer(unsigned(x_exp));
       num_y_exp <= conv_integer(unsigned(y_exp));


    -- Membandingkan exponen dan shifting mantissa ------------------
       if (x_exp > y_exp) then
          dif_exp <= num_x_exp - num_y_exp - 1;
          x_man_up <= x_man;
          SS <= x_sign;


             if (dif_exp = 0) then
                y_man_up(9) <= '1';
                y_man_up(8 downto 0) <= y_man(9 downto 1);
             else
                for j in 9 downto 0 loop
                   y_man_up(j) <= '0';
                end loop;
                y_man_up(9-dif_exp) <= '1';
                y_man_up(8-dif_exp downto 0) <= y_man(9 downto 1+dif_exp);
             end if;


          elsif (x_exp < y_exp) then
             dif_exp <= num_y_exp - num_x_exp - 1;
             y_man_up <= y_man;
             SS <= y_sign;

             if (dif_exp = 0) then
                x_man_up(9) <= '1';
                x_man_up(8 downto 0) <= y_man(9 downto 1);
             else
                for j in 9 downto 0 loop
                   x_man_up(j) <= '0';
                end loop;
                x_man_up(9-dif_exp) <= '1';
                x_man_up(8-dif_exp downto 0) <= y_man(9 downto 1+dif_exp);
             end if;


          elsif (x_exp = y_exp) then
             x_man_up <= x_man;
             y_man_up <= y_man;


          end if;


       end if;
    end process;
```

Apabila kasus normal/subnormal, dilakukan pembandingan besar exponent, lalu dilakukan shifting apabila nilai exponent berbeda.

```
    -- Operasi +/- pada mantissa -----------------------

    process (x_sign,y_sign,norm)
    begin
       if (x_sign = '1' and norm = '1') then
          if (y_sign = '0') then
             operator <= x_sign;
             x_op <= y_man_up(9 downto 0);
             y_op <= x_man_up(9 downto 0);
          else
             operator <= '0';
             x_op     <= x_man_up(9 downto 0);
             y_op     <= y_man_up(9 downto 0);
          end if;


       elsif (x_sign = '0' and norm = '1') then
          if (y_sign = '1') then
             operator <= y_sign;
             x_op <= x_man_up(9 downto 0);
             y_op <= y_man_up(9 downto 0);
          else
             operator <= '0';
             x_op     <= x_man_up(9 downto 0);
             y_op     <= y_man_up(9 downto 0);
          end if;

    end if;
    end process;
```

Karena menggunakan adder substractor, mantissa yang akan dikurangi (dijadikan negatif) diletakkan pada urutan operasi kedua. Mantissa yang diubah menjadi negatif adalah mantissa pada floating point dengan sign 1 (negatif).

```
    -- Operasi pada mantissa --------------
    comp : adder_substractor
       port map(operator, x_op, y_op, MS2, cout);


    -- Keluaran/output ----------------------
    output(15) <= SS;
    output(14 downto 10) <= ES;
    output(9 downto 0) <= MS when norm ='0' else
                          MS2;

end architecture structural;
```

Terakhir, dilakukan operasi pada mantissa apabila normal, kemudian keluaran akhir ditaruh pada port output. Untuk output mantissa menggunakan MS apabila kasus spesial dan menggunakan MS 2 apabila kasus subnormal/normal.

Hasil *Run RTL* didapat sebagai berikut.