Nama : Sidatha Prastya. P
NIM : 13219033

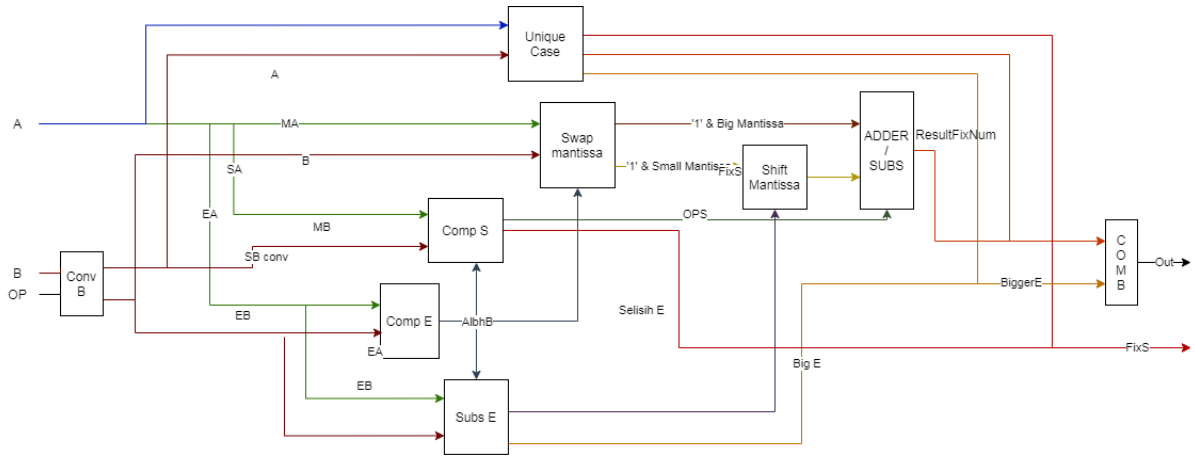
Laporan Praktikum 3C
EL 2102 – Praktikum Sistem Digital
Floating Point Adder

I. Spesifikasi

Pada laporan praktikum kali ini ditugaskan untuk membuat desain sekaligus VHDL *code* mengenai *Half Precision Floating Point Adder* sesuai dengan standar IEEE 754. Laporan ini merupakan kelanjutan dari laporan sebelumnya yaitu mengenai desain rangkaian dari *adder* ini. Pada project ini terdapat beberapa file yang merupakan komponen pembentuk pada *floating point adder*.

II. Perancangan

Pada praktikum ini, ditugaskan untuk membuat sistem operasi half-point precision. Praktikum ini merupakan kelanjutan dari praktikum sebelumnya dengan tujuan memperbaiki sekaligus membuat desain dari half-precision floating point dengan standar IEEE 754. Berikut adalah diagram alur dari operasi hitung half-precision floating point.

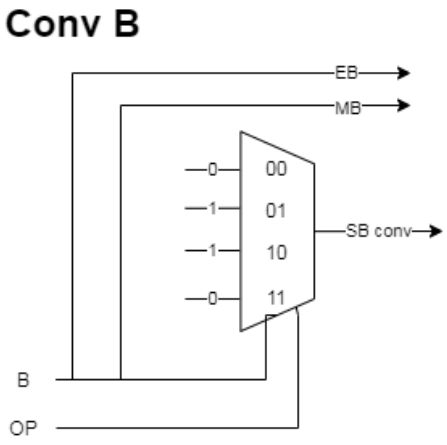


Berikut adalah penjelasan dari masing-masing tahapan.

1. Conv B

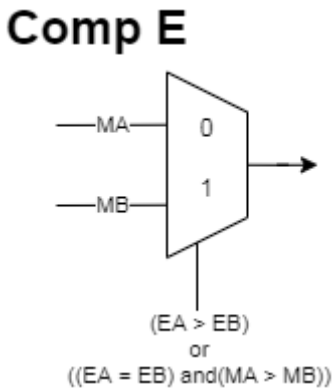
Conv B merupakan logika untuk mengubah sign B terhadap operator yang dimasukkan. Apabila OP = ‘0’ berarti dilakukan penambahan, sedangkan apabila sebaliknya, maka dilakukan pengurangan. Sign B akan berubah mengikuti operator yang ada. Berikut diagram Conv B.

Nama : Sidatha Prastya. P
NIM : 13219033



2. Comp E

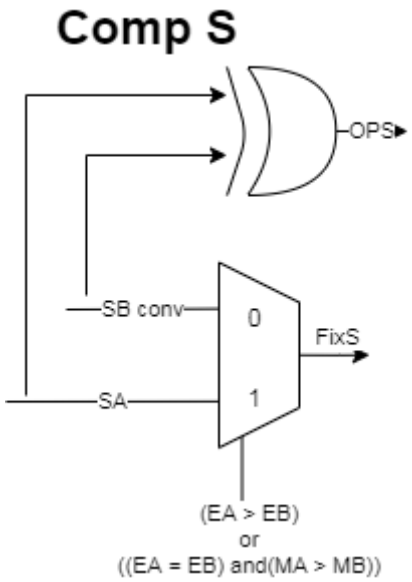
Comp E merupakan logika untuk membandingkan eksponen pada A dan pada B. Apabila ekponen A lebih besar daripada eksponen B, maka akan menghasilkan variabel AlbhB bernilai ‘1’ dan juga sebaliknya akan menghasilkan ‘0’. Akan tetapi, apabila eksponen A sama dengan eksponen B, maka akan dilakukan pemeriksaan mantissa. Apabila mantissa A lebih besar daripada mantissa B, maka akan menghasilkan AlbhB bernilai ‘1’ begitu juga sebaliknya. Berikut diagram CompE.



3. Comp S

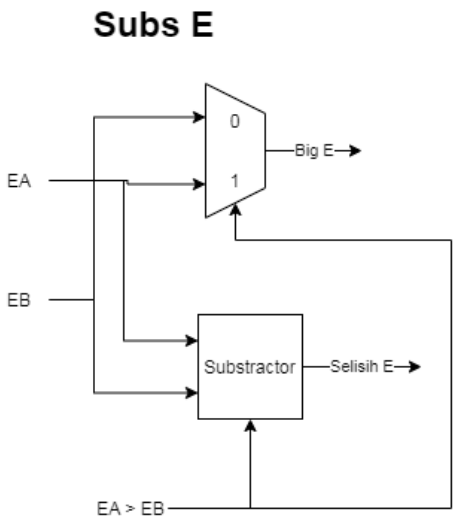
Comp S merupakan logika untuk membandingkan nilai sign pada A dan B. Sign B yang digunakan adalah keluaran dari conv B. Untuk parameter, Comp S menggunakan output dari Comp E. Comp S akan menentukan operasi pengurangan atau penambahan pada mantissa, sekaligus mengeluarkan hasil akhir sign pada keluaran akhir. Berikut diagram Comp S.

Nama : Sidatha Prastya. P
NIM : 13219033



4. Subs E

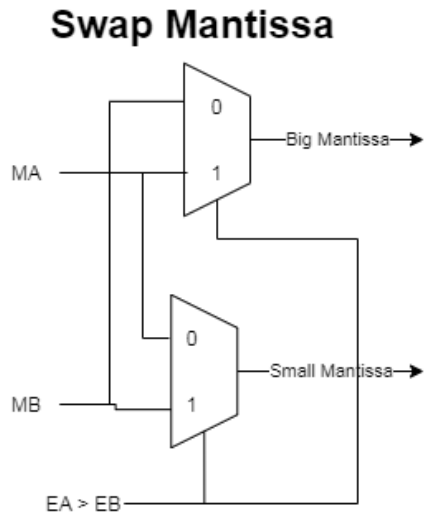
Subs E merupakan logika untuk melakukan pengurangan pada eksponen A dan eksponen B. Output yang dihasilkan berupa selisih eksponen dan juga sekaligus menentukan eksponen terbesar. Berikut diagram Subs E.



5. Swap Mantissa

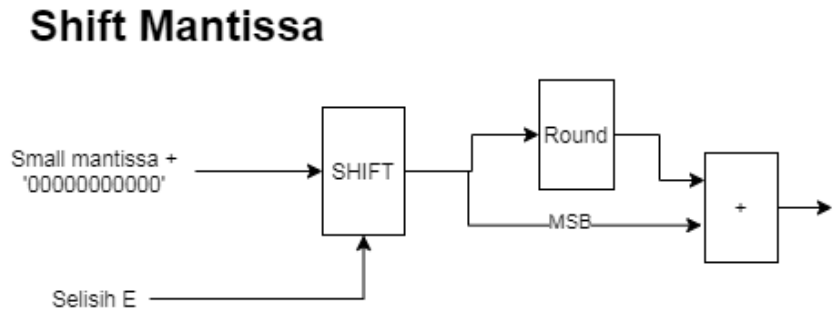
Swap mantissa merupakan logika untuk menentukan mantissa besar dan matissa kecil. Tujuannya adalah akan dilakukan shifting pada mantissa kecil sebesar selisih eksponen hasil output Subs E. berikut adalah diagram Swap Mantissa.

Nama : Sidatha Prastya. P
NIM : 13219033



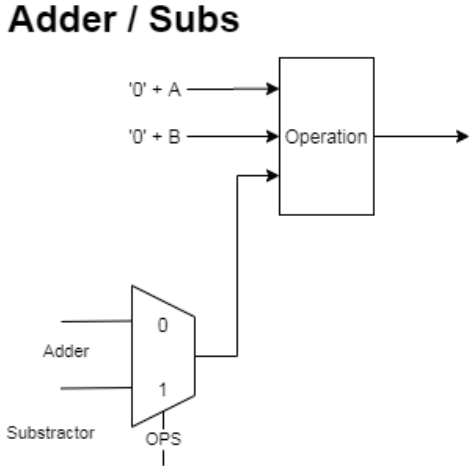
6. Shift Mantissa

Shift mantissa merupakan logika untuk melakukan shifting pada mantissa sebesar selisih antara eksponen A dan eksponen B. Berikut diagram Shift mantissa.



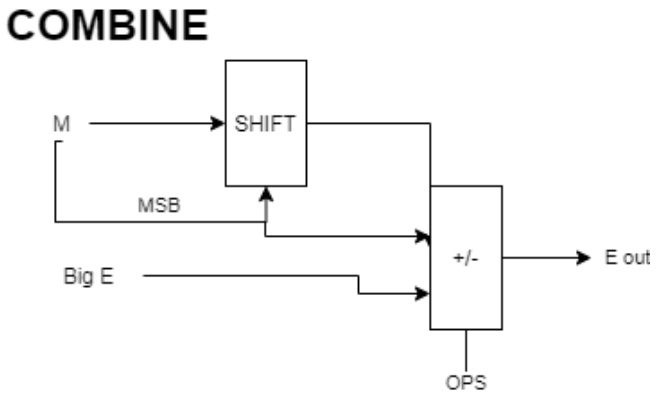
7. Adder/Subs

Adder/Subs merupakan operasi penambahan dan pengurangan pada mantissa. Pada penentuan penambahan/pengurangan menggunakan OPS sebagai penentu. OPS merupakan keluaran dari Comp S. Keluaran dari logika ini adalah ResultFixNum. Berikut diagram Adder/Subs.



8. Combine

Combine merupakan logika untuk menggabungkan hasil akhir dari operasi mantissa dan exponen terbesar. Pertama, mantissa hasil operasi akan diambil berdasarkan MSB atau carry hasil penjumlahan/pengurangan. MSB ini akan ditambahkan ke eksponen output. Logika ini akan menghasilkan mantissa 10 bit dan eksponen 5 bit. Berikut diagram dari Combine.



III. Pembahasan

Dalam proses pemrograman terdapat beberapa file VHDL yang berperan sebagai kompnen-komponen pembentuk, yaitu:

1. Adder11.vhd

Komponen ini merupakan bagian yang berguna untuk melakukan operasi penjumlahan atau pengurangan pada mantissa.

Input:

- InMSatu : input mantissa pertama (11 bit).
- InMDua : input mantissa kedua (11 bit).
- OPS : penentu operasi yang bernilai ‘0’ bila penjumlahan atau ‘1’ bila pengurangan (1 bit).

Output:

- OutM : output mantissa hasil operasi (12 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity Adder11 is
```

```
port
      (
            -- Input ports
            InMSatu     : in  unsigned (10 downto 0);
            InMDua      : in  unsigned (10 downto 0);
            OPS         : in std_logic;
            -- Output ports
            OutM  : out unsigned (11 downto 0)
      );
end Entity;

Architecture RTL of Adder11 is
Begin
      OutM <= ('0' & InMSatu) + ('0' & InMDua) when (OPS =
                  '0') else
               ('0' & InMSatu) - ('0' & InMDua);
End RTL;
```

2. Combine.vhd

Komponen ini merupakan bagian untuk menggabungkan mantissa beserta eksponen. Pada bagian ini juga terjadi pembulatan mantissa hasil operasi hitung.

Input:

- ResultFixNum : mantissa hasil operasi adder (12 bit).
- BiggerE : Nilai eksponen terbesar (5 bit).
- op : operasi yang terjadi pada adder, ‘0’ bila penjumlahan atau ‘1’ bila pengurangan (1 bit).

Output:

- ResultME : hasil penggabungan mantissa dan eksponen (15 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity Combine is
      port
      (
            -- Input ports
            ResultFixNum: in  unsigned (11 downto 0);
            BiggerE     : in  unsigned (4 downto 0);
            op          : in std_logic;

            -- Output ports
            ResultME    : out unsigned (14 downto 0)
      );

end Entity;
```

```
Architecture RTL of Combine is
      Signal Exponent : Unsigned(4 Downto 0);
Begin
      Exponent <= (BiggerE + 1) when (ResultFixNum(11) =
                        '1' and op ='0') else
                  (BiggerE - 1) when (ResultFixNum(11) = '1'
                        and op ='1') else
                  BiggerE;

      ResultME(14 Downto 10) <= exponent;
      ResultME(9 Downto 0) <= ResultFixNum(10 Downto 1)
            when ResultFixNum(11) = '1' else ResultFixNum(9
            Downto 0);

End RTL;
```

3. CompS.vhd

Komponen ini berfungsi untuk membandingkan sign A dan sign B (hasil konversi) dengan parameter A>B.

Input:

- SA : sign input A (1 bit).
- SBConv : sign input B hasil konversi (1 bit).
- AlbhB : parameter yang menyatakan bila A>B (1 bit).

Output:

- OPS : bilangan bernilai 1 atau 0 yang menentukan penjumlahan atau pengurangan (1 bit).
- FixS : sign yang menjadi output akhir dari floating point adder (1 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity CompS is
      port
      (
            -- Input ports
            SA    : in  std_logic;
            SBconv      : in  std_logic;
            AlbhB : in  Std_logic;

            -- Output ports
            OPS   : out std_logic;
            FixS: out std_logic
      );
```

```
end entity;

Architecture RTL of CompS is

Begin
      FixS <= SA when (AlbhB = '1') else
                        SBconv;
      OPS <= SA xor SBconv;

End RTL;
```

4. ConvB.vhd

Komponen ini berfungsi untuk mengubah sign input B terhadap operator yang diberikan pertama kali.

Input:

- BOP : sign B sebelum diubah terhadap operator (1 bit).
- OP : operator pada saat pertama kali memasukkan nilai, bernilai ‘0’ bila pengurangan atau ‘1’ bila penjumlahan (1 bit).

Output:

- BConv : sign B setelah dikonversi terhadap operator (1 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity ConvB is
      port
      (
            -- Input ports
            BOP   : in  std_logic;
            OP    : in  std_logic;

            -- Output ports
            Bconv : out std_logic
      );
end entity;

Architecture RTL of ConvB is

Begin

      Bconv <= BOP xor OP;

End RTL;
```

Nama : Sidatha Prastya. P
NIM : 13219033

5. ShiftR.vhd

Komponen ini berfungsi untuk melakukan pergeseran mantissa sebesar selisih eksponen A dan B.

Input:

- InM : mantissa yang akan dilakukan *shifting* (11 bit).
- SelisE : selisih dari eksponen A dan B (5 bit).

Output:

- OutM : mantissa hasil *shifting* (11 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity ShiftR is
      port
      (
            -- Input ports
            InM   : in  unsigned (10 downto 0);
            SelisE      : in  unsigned (4 downto 0);

            -- Output ports
            OutM  : out unsigned (10 downto 0)
      );
end Entity;

Architecture RTL of ShiftR is
      Signal Shifted_Signal : unsigned(21 downto 0);
      Signal Internal_Signal : std_logic_vector(9 downto
      0);
      Signal Round_Up : std_logic;

      Begin

      with SelisE select
      Shifted_Signal <= (InM & "00000000000") when "00000",
            ("0" &  InM & "0000000000") when "00001",
            ("00" &  InM & "000000000") when "00010",
            ("000" &  InM & "00000000") when "00011",
            ("0000" &  InM & "0000000") when "00100",
            ("00000" &  InM & "000000") when "00101",
            ("000000" &  InM & "00000") when "00110",
            ("0000000" &  InM & "0000") when "00111",
            ("00000000" &  InM & "000") when "01000",
            ("000000000" &  InM & "00") when "01001",
            ("0000000000" &  InM & "0") when "01010",
            ("00000000000" &  InM) when "01011",
      ("000000000000" &  InM(10 Downto 1)) when "01100",
      "00000000000000000000" when others;
```

```
      OutM <= Shifted_Signal(21 Downto 11) when (Round_Up =
            '0') else
            (Shifted_Signal(21 Downto 11) + 1);

      Internal_Signal <= std_logic_vector(Shifted_Signal(9
            Downto 0));
      Round_Up <= Shifted_Signal(10) when (Internal_Signal
            /= "000000000") else
            '0';

End RTL;
```

6. SubsE.vhd

Komponen ini berfungsi untuk mendapatkan selisih dari eksponen A dan eksponen B. Selain itu, komponen ini juga berfungsi untuk mendapatkan nilai eksponen terbesar sesuai dengan parameter A>B.

Input:

- EA : eksponen floating point A (5 bit).
- EB : eksponen floating point B (5 bit).
- AlbhB : parameter yang menyatakan bila A>B (1 bit).

Output:

- SelisE : selisih dari eksponen A dan eksponen B (5 bit).
- BiggerE : nilai eksponen terbesar (5 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity SubsE is
      port
      (
            -- Input ports
            EA    : in  unsigned (4 downto 0);
            EB    : in  unsigned (4 downto 0);
            AlbhB : in  Std_logic;

            -- Output ports
            SelisE      : out unsigned (4 downto 0);
            BiggerE: out unsigned (4 downto 0)
      );
end entity;
```

```
Architecture RTL of SubsE is

Begin

      BiggerE <= EA when (AlbhB = '1') else
                              EB;

      SelisE <= (EA - EB) when (AlbhB = '1') else
                              (EB-EA);

End RTL;
```

7. SwapM.vhd

Komponen ini berfungsi untuk memilah nilai mantissa yang lebih besar dan lebih kecil dengan menggunakan parameter A>B. Mantissa yang lebih kecil akan dilakukan *shifting* oleh komponen ShiftR.

Input:

- MA : input mantissa A (10 bit).
- MB : input mantissa B (10 bit).
- AlbhB : parameter yang menyatakan bila A>B (1 bit).

Output:

- BiggerI_Mantissa : mantissa yang lebih besar (11 bit).
- SmallerI_Mantissa : mantissa yang lebih kecil (11 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

Entity SwapM is
      port
      (
            -- Input ports
            MA    : in  unsigned (9 downto 0);
            MB    : in  unsigned (9 downto 0);
            AlbhB : in  Std_logic;

            -- Output ports
            BiggerI_Mantisa   : out unsigned (9 downto 0);
            SmallerI_Mantisa  : out unsigned (9 downto 0)
      );
end Entity;
```

```
Architecture RTL of SwapM is
Begin

      BiggerI_Mantisa <= MA when (AlbhB = '1') else
                         MB;
      SmallerI_Mantisa <= MA when (AlbhB = '0') else
                         MB;

End RTL;
```

8. FPAdder.vhd

Komponen ini merupakan komponen utama yang menggabungkan seluruh komponen di atas. Ini merupakan program utama untuk *Half Precision Floating Point Adder*.

Input:

- a : input floating point pertama (16 bit).
- b : input floating point kedua (16 bit).
- op : operator perhitungan. 0 bila penjumlahan dan 1 bila pengurangan (1 bit).

Output:

- result : hasil akhir dari penjumlahan/pengurangan floating point (16 bit).

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;


entity FPadder is
      port
      (
            a        : in std_logic_vector  (15 downto 0);
            b        : in std_logic_vector  (15 downto 0);
            op          : in std_logic;
            result : out std_logic_vector  (15 downto 0)
      );

end entity;

architecture structural of FPadder is

component ConvB
      port
      (
            -- Input ports
            BOP   : in  std_logic;
            OP          : in  std_logic;
```

```
            -- Output ports
            Bconv : out std_logic
      );
end component;

component CompS
      port
      (
            -- Input ports
            SA                : in  std_logic;
            SBconv      : in  std_logic;
            AlbhB       : in  Std_logic;

            -- Output ports
            OPS   : out std_logic;
            FixS  : out std_logic
      );
end component;

component SwapM
      port
      (
            -- Input ports
            MA    : in  unsigned (9 downto 0);
            MB    : in  unsigned (9 downto 0);
            AlbhB : in  std_logic;

            -- Output ports
            BiggerI_Mantisa   => BiggerI_Mantisa,
            SmallerI_Mantisa  => SmallerI_Mantisa
      );

-- Block Comparator
      AlbhB <= '1' when ((EA > EB) or ((EA = EB) and (MA >
            MB))) else
                        '0';

-- Instansiasi Komponen SubsE
Komponen_Subse : SubsE
      port map
      (
            EA,
            EB,
            AlbhB,
            SelisE,
            BiggerE
      );

-- Add '1' to mantisa
SmallFixNum <= '1' & SmallerI_Mantisa;
BigFixNum <= '1' & BiggerI_Mantisa;
```

```
-- Instansiasi Komponen ShiftR
Komponen_ShiftR : ShiftR
      port map
      (
            -- Input ports
            InM   => SmallFixNum,
            SelisE      => SelisE,

            -- Output ports
            OutM  => ShiftedFixNum
      );

-- Instansiasi Komponen Adder untuk mantisa
Komponen_Adder11 : Adder11
      port map
      (
        ShiftedFixNum,
        BigFixNum,
        OPS,

            -- Output ports
        ResultFixNum
      );

-- Instansiasi Komponen Combine untuk memggabungkan hasil
akhir dengan exponent
Komponen_combine: Combine
      port map
      (
            -- Input ports
            ResultFixNum,
            BiggerE,
            OPS,

            -- Output ports
            ResultME`
      );

Result <= std_logic_vector(FixS & ResultME);

end structural;
```

IV. Pengujian

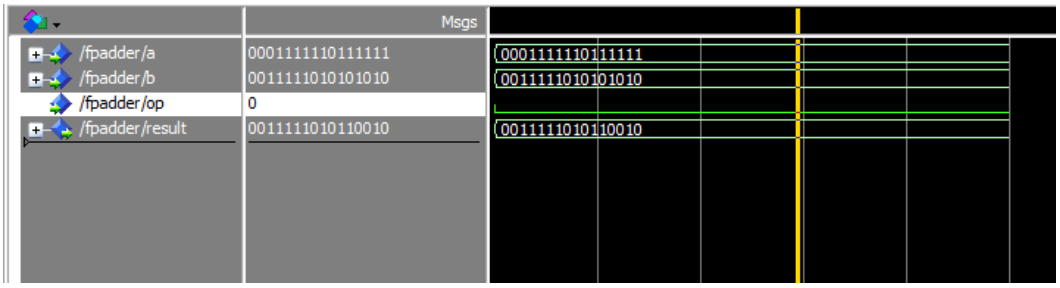
Akan dilakukan pengujian terhadap desain dan program yang telah dibuat. Pada simulasi pertama, akan dilakukan penjumlahan kedua floating point positif yang akan menghasilkan floating point positif.

Input : 0 00111 1110111111 dan 0 01111 1010101010.

Operasi : penjumlahan positif dan positif

Output ekspektasi: 0 01111 1010110010.

Simulasi:

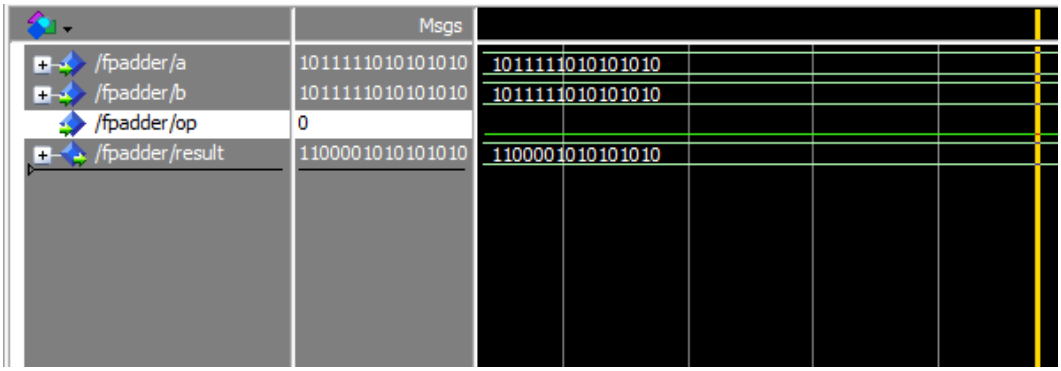


Uji penjumlahan floating point negative dan negative.

Input : 1 00111 1110111111 dan 1 00111 1110111111

Operasi : penjumlahan negative dan negatif

Simulasi:



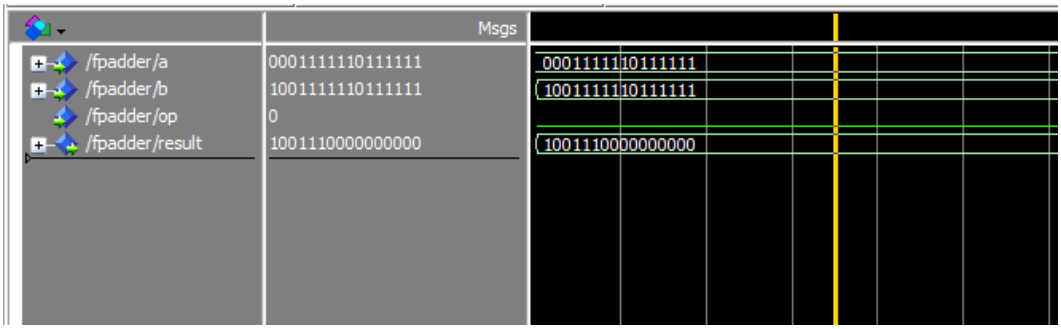
Lalu, dilakukan uji apabila terjadi penjumlahan antara bilangan positif dan negatif dengan besar yang sama sehingga menghasilkan mantissa bernilai 0.

Input : 0 00111 1110111111 dan 1 00111 1110111111

Operasi : penjumlahan positif dan negative dengan besar sama.

Output ekspektasi: 0 00111 0000000000 atau 1 00111 0000000000

Simulasi:

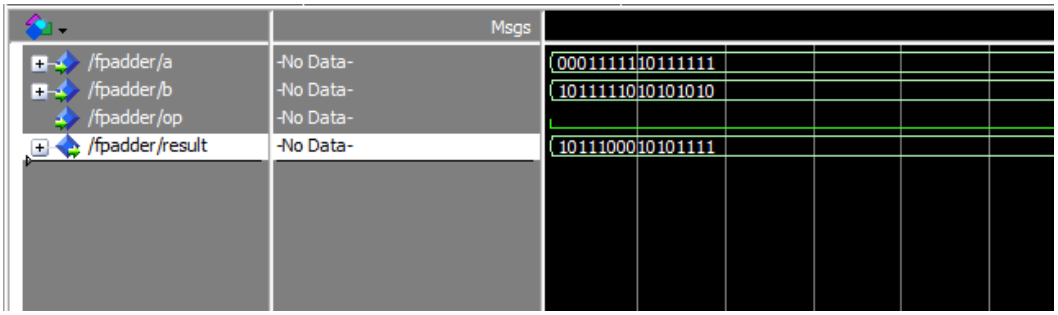


Untuk pengujian selanjutnya dilakukan penjumlahan antara bilangan positif dan negatif dengan besar yang berbeda. Berikut adalah hasil dari simulasi program. Pertama, dilakukan penjumlahan floating point positif dengan negative.

Input : 0 00111 1110111111 dan 1 01111 1010101010

Operasi : penjumlahan positif dan negatif

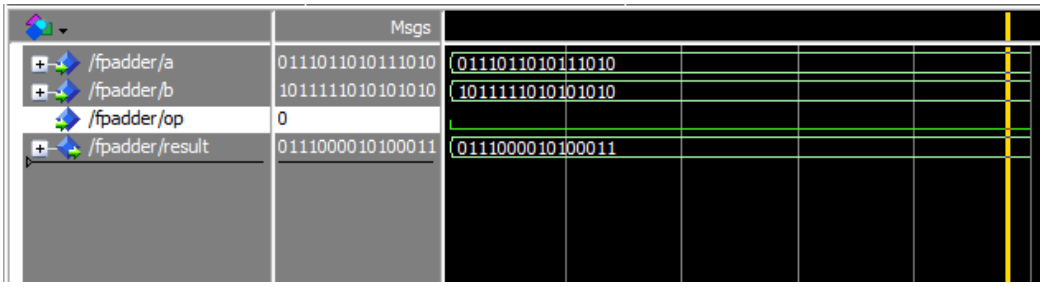
Simulasi : (hasil negative)



Input : 0 11101 1010111010 dan 1 01111 1010101010

Operasi : penjumlahan positif dan negatif

Simulasi : (hasil positif)

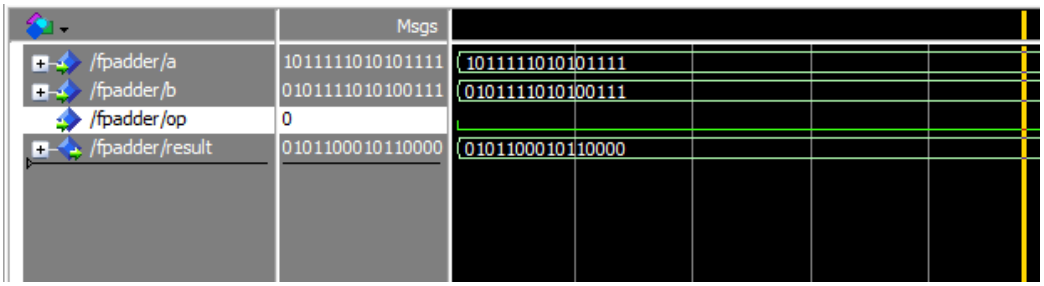


Untuk kasus selanjutnya, dilakukan penjumlahan antara bilangan negative dan bilangan positif.

Input : 1 01111 1010101111 dan 0 10111 1010100111

Operasi : penjumlahan negative dan positif

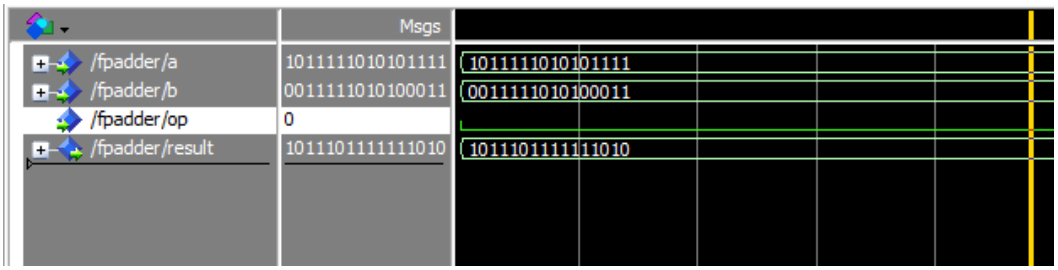
Simulasi : (hasil positif)



Input : 1 01111 1010101111 dan 0 01111 1010100011

Operasi : penjumlahan negative dan positif

Simulasi : (hasil negative)



Lalu, simulasi terakhir dengan sign yang berbeda, namun dengan operasi pengurangan yang menghasilkan floating adder positif

Input : 0 00111 1110111111 dan 1 00111 1110001111

Operasi : pengurangan positif dan negative

Simulasi :