MINI-PROJECT - 1

(IEEE 754) FLOATING POINT ADDER UN-PIPELINED

CODE:

```
#include<stdio.h>
#include<conio.h>

void main()
 {
        int A_sign_18[6]={ 0 , 0 , 1 , 1 , 0 , 0},              //Array of Signs of A
            B_sign_18[6]={ 0 , 1 , 0 , 1 , 0 , 1},              //Array of Signs of B
            Sum_sign_18[6],                                     //Array of Signs of Sum
            A_exp_18[6]={ 0x85 , 0x85 , 0x84 , 0x87 , 0x00 , 0x00 }, //Array of Exponents of A
            B_exp_18[6]={ 0x86 , 0x85 , 0x85 , 0x85 , 0x00 , 0x85 },      //Array of Exponents of B
            Sum_exp_18[6],                                      //Array of Exponents of Sum
            shift,                                              //Shifts required to align Mantissa
            i=0;                                                //Index of elements of array

                    //Array of Mantissa's of A
   unsigned long int A_mantissa_18[6]={ 0x460000, 0x420000, 0x5c0000, 0x0c0000, 0x000000,
0x000000 },
                    //Array of Mantissa's of B
                   B_mantissa_18[6]={ 0x320000, 0x1e0000, 0x160000, 0x0a0000, 0x000000,
0x620000 },
                    //Array of Mantissa's of Sum
                    Sum_mantissa_18[6],
                    //Masks for Normalization
                    mask1=0x400000,
                    mask2=0x7fffff,
                    //Array of Normalized Sum
                    x[6];

for(i=0;i<6;i++)                        //Loop to access each element of the array sequentially
        {
                printf("Testbench %d\n",i+1);
                printf("A is %d %x %lx\n",A_sign_18[i],A_exp_18[i],A_mantissa_18[i]);
                printf("B is %d %x %lx\n",B_sign_18[i],B_exp_18[i],B_mantissa_18[i]);

printf("\n------------------------------Comparison of Exponent's--------------------------------");

                //Loop to determine the shift and the greater number

                if(A_exp_18[i]>B_exp_18[i])
                {
                        shift=A_exp_18[i]-B_exp_18[i];
```

```
                printf("\nA is greater\n");
        }
        else if(A_exp_18[i]<B_exp_18[i])
        {
                shift=B_exp_18[i]-A_exp_18[i];
                printf("\nB is greater\n");
        }
        else
        {
                shift=0;
        }

//Including the 1 from "1.m" into the Mantissa
        A_mantissa_18[i]=A_mantissa_18[i]>>1;
        B_mantissa_18[i]=B_mantissa_18[i]>>1;
        A_mantissa_18[i]=A_mantissa_18[i]|mask1;
        B_mantissa_18[i]=B_mantissa_18[i]|mask1;
        A_exp_18[i]+=1;
        B_exp_18[i]+=1;
        printf("\nExtra shifted no's are \n%d %x %lx\n%d %x %lx", A_sign_18[i], A_exp_18[i],
A_mantissa_18[i], B_sign_18[i], B_exp_18[i], B_mantissa_18[i]);
        printf("\nNo. of shifts = %d\n",shift);

printf("\n----------Right Shifting of Mantissa's----------");

//Aligning the Mantissa based on the number of shifts required and making the exponents same
        if(A_exp_18[i]<B_exp_18[i])
        {
                A_mantissa_18[i]=A_mantissa_18[i]>>shift;
                A_exp_18[i]=A_exp_18[i]+shift;
                Sum_sign_18[i]=B_sign_18[i];
        }
        else if(A_exp_18[i]>B_exp_18[i])
        {
                B_mantissa_18[i]=B_mantissa_18[i]>>shift;
                B_exp_18[i]=B_exp_18[i]+shift;
                Sum_sign_18[i]=A_sign_18[i];
        }
        else
        {
                A_mantissa_18[i]=A_mantissa_18[i];
                B_mantissa_18[i]=B_mantissa_18[i];
                Sum_sign_18[i]=A_sign_18[i];
        }
        Sum_exp_18[i]=A_exp_18[i];
    printf("\nAlligned no. A is%d %x %lx\n",A_sign_18[i],A_exp_18[i],A_mantissa_18[i]);
printf("\nAlligned mantissa of B is %d %x %lx\n", B_sign_18[i], B_exp_18[i], B_mantissa_18[i]);
```

```
printf("\n-----------------------------------Alignment-----------------------------------");

		//Determining the smaller mantissa and taking it's 2's complement
		if(A_sign_18[i]!=B_sign_18[i])
		{
			if(A_mantissa_18[i]<B_mantissa_18[i])
			{
				A_mantissa_18[i]=(~A_mantissa_18[i])+1;
			}
			else
			{
				B_mantissa_18[i]=(~B_mantissa_18[i])+1;
			}
	printf("\nMantissa's after 2's complement %lx %lx\n", A_mantissa_18[i], B_mantissa_18[i]);
		}
		else
		{
			A_mantissa_18[i]=A_mantissa_18[i];
			B_mantissa_18[i]=B_mantissa_18[i];
			printf("\n2's complement is not required as signs are same\n");
		}

printf("\n-----------------------------------Addition-----------------------------------");

		//Addition of the Mantissa's
		Sum_mantissa_18[i]=A_mantissa_18[i]+B_mantissa_18[i];
	printf("\nSum is %d %x %lx\n" , Sum_sign_18[i] ,Sum_exp_18[i],Sum_mantissa_18[i]);

printf("\n---------------------------------Normalization---------------------------------");

		//Normalization of the Mantissa's

if(A_sign_18[i]!=B_sign_18[i])			//For Sign(A) != Sign(B)
		{
			if(A_sign_18[i]==0&B_sign_18[i]==1)		//For A=+ve & B=-ve
			{
				if(A_mantissa_18[i]==0)		//For A=0
				{
					Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
					x[i]=Sum_mantissa_18[i]&mask2;
					x[i]=x[i]>>1;
					Sum_exp_18[i]=Sum_exp_18[i]-1;
	printf("\nNormalized Value of Sum is %d %x %lx\n" , Sum_sign_18[i], Sum_exp_18[i], x[i]);
				}
```

```
                        else                                    //For A !=0
                        {
                                Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
                                x[i]=Sum_mantissa_18[i]&mask2;
                                Sum_exp_18[i]=Sum_exp_18[i]-2;
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i], Sum_exp_18[i],x[i]);
                        }
                }
   else                                         //For Sign(A) != Sign(B)
                {
                        Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
                        x[i]=Sum_mantissa_18[i]&mask2;
                        Sum_exp_18[i]=Sum_exp_18[i]-2;
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i], Sum_exp_18[i], x[i]);
                }
        }
else                                                    //For Sign(A) = Sign(B)
        {
                if(A_sign_18[i]==0&B_sign_18[i]==0)             //Both A & B = +ve
                {
                        if(A_mantissa_18[i]==0)         //For A=0
                        {
                                x[i]=Sum_mantissa_18[i]&mask2;
                                Sum_exp_18[i]=Sum_exp_18[i]-1;
                        }
                        else                                    //For A != 0
                        {
                                x[i]=Sum_mantissa_18[i]&mask2;
                        }
                }
                else if(A_sign_18[i]==1&B_sign_18[i]==1)        //Both A & B = -ve
                {
                        Sum_mantissa_18[i]=Sum_mantissa_18[i]<<1;
                        Sum_exp_18[i]=Sum_exp_18[i]-1;
                        x[i]=Sum_mantissa_18[i]&mask2;
                }
                else{}
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i], Sum_exp_18[i],x[i]);
                }
           getch();
        }
 }
```

OUTPUT:

FOR A = 99, B = 178

```
A is 0 85 460000
B is 0 86 320000

------------------------------Comparison of Exponent's--------------------------------
B is greater

Extra shifted no's are
0 86 630000
0 87 590000
No. of shifts = 1

----------------------------Right Shifting of Mantissa's----------------------------
Alligned no. A is0 87 318000

Alligned mantissa of B is 0 87 590000

-------------------------------------Alignment--------------------------------------
2's complement is not required as signs are same

-------------------------------------Addition---------------------------------------
Sum is 0 87 8a8000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 0 87 a8000
```

FOR A = 97, B = -79

```
                                    Testbench 2
A is 0 85 420000
B is 1 85 1e0000

------------------------------Comparison of Exponent's--------------------------------
Extra shifted no's are
0 86 610000
1 86 4f0000
No. of shifts = 0

----------------------------Right Shifting of Mantissa's----------------------------
Alligned no. A is0 86 610000

Alligned mantissa of B is 1 86 4f0000

-------------------------------------Alignment--------------------------------------
Mantissa's after 2's complement 610000 ffb10000

-------------------------------------Addition---------------------------------------
Sum is 0 86 120000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 0 84 480000
```

FOR A = -55, B = 75

```
Testbench 3
A is 1 84 5c0000
B is 0 85 160000

------------------------------Comparison of Exponent's--------------------------------
B is greater

Extra shifted no's are
1 85 6e0000
0 86 4b0000
No. of shifts = 1

-----------------------------Right Shifting of Mantissa's-----------------------------
Alligned no. A is1 86 370000

Alligned mantissa of B is 0 86 4b0000

-------------------------------------Alignment--------------------------------------
Mantissa's after 2's complement ffc90000 4b0000

-------------------------------------Addition---------------------------------------
Sum is 0 86 140000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 0 84 500000
```

FOR A = -280, B = -69

```
Testbench 4
A is 1 87 c0000
B is 1 85 a0000

------------------------------Comparison of Exponent's--------------------------------
A is greater

Extra shifted no's are
1 88 460000
1 86 450000
No. of shifts = 2

-----------------------------Right Shifting of Mantissa's-----------------------------
Alligned no. A is1 88 460000

Alligned mantissa of B is 1 88 114000

-------------------------------------Alignment--------------------------------------
2's complement is not required as signs are same

-------------------------------------Addition---------------------------------------
Sum is 1 88 574000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 1 87 2e8000
```

FOR A = 0, B = 0

```
Testbench 5
A is 0 0 0
B is 0 0 0

------------------------------Comparison of Exponent's--------------------------------
Extra shifted no's are
0 1 400000
0 1 400000
No. of shifts = 0

----------------------------Right Shifting of Mantissa's----------------------------
Alligned no. A is0 1 400000

Alligned mantissa of B is 0 1 400000

-------------------------------------Alignment--------------------------------------
2's complement is not required as signs are same

-------------------------------------Addition---------------------------------------
Sum is 0 1 800000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 0 1 0
```

FOR A = 0, B = -113

```
Testbench 6
A is 0 0 0
B is 1 85 620000

------------------------------Comparison of Exponent's--------------------------------
B is greater

Extra shifted no's are
0 1 400000
1 86 710000
No. of shifts = 133

----------------------------Right Shifting of Mantissa's----------------------------
Alligned no. A is0 86 20000

Alligned mantissa of B is 1 86 710000

-------------------------------------Alignment--------------------------------------
Mantissa's after 2's complement fffe0000 710000

-------------------------------------Addition---------------------------------------
Sum is 1 86 6f0000

-----------------------------------Normalization------------------------------------
Normalized Value of Sum is 1 84 3c0000
```

MINI-PROJECT - 1

(IEEE 754) FLOATING POINT ADDER WITH 5 – STAGE PIPELINE

CODE:

```
#include<stdio.h>
#include<conio.h>

 void Compare(int i);
 void Right_Shift(int i);
 void Align(int i);
 void Add(int i);
 void Normalize(int i);

        int A_sign_18[6]={ 0 , 0 , 1 , 1 , 0 , 0 },                    //Array of Signs of A
            B_sign_18[6]={ 0 , 1 , 0 , 1 , 0 , 1 },                    //Array of Signs of B
            Sum_sign_18[6],                                            //Array of Signs of Sum
            A_exp_18[6]={ 0x85 , 0x85 , 0x84 , 0x87 , 0x00 , 0x00 },   //Array of Exponents of A
            B_exp_18[6]={ 0x86 , 0x85 , 0x85 , 0x85 , 0x00 , 0x85 },   //Array of Exponents of B
            Sum_exp_18[6],                                             //Array of Exponents of Sum
            shift;                                                     //Shifts required to align Mantissa

        short int clock;                                               //To simulate a Sequential execution

                                //Array of Mantissa's of A
        unsigned long int A_mantissa_18[6]={ 0x460000 , 0x420000 , 0x5c0000 , 0x0c0000 ,
0x000000 ,  0x000000 },
                                //Array of Mantissa's of B
                                B_mantissa_18[6]={ 0x320000 , 0x1e0000 , 0x160000 , 0x0a0000 ,
0x000000 , 0x620000 },
                                //Array of Mantissa's of Sum
                                Sum_mantissa_18[6],
                                //Masks for Normalization
                                mask1,
                                mask2,
                                //Array of Normalized Sum
                                x[6];

 void main()
 {
        for(clock=0;clock<10;clock++)
                {
                        switch(clock)
```

```
{
        case 0:
        {
                Compare(0);                     //Comparison of Pair 1
                printf("\nEND OF STAGE 0");
                    break;
        }

        case 1:
        {
                Right_Shift(0);         //Right Shifting of Pair 1
                Compare(1);                     //Comparison of Pair 2
                printf("\nEND OF STAGE 1");
                break;
        }
        case 2:
        {
                Align(0);                       //Alignment of Pair 1
                Right_Shift(1);         //Right Shifting of Pair 2
                Compare(2);                     //Comparison of Pair 3
                printf("\nEND OF STAGE 2");
                break;
        }
        case 3:
        {
                Add(0);                         //Addition of Pair 1
                Align(1);                       //Alignment of Pair 2
                Right_Shift(2);         //Right Shifting of Pair 3
                Compare(3);                     //Comparison of Pair 4
                printf("\nEND OF STAGE 3");
                break;
        }
        case 4:
        {
                Normalize(0);                   //Normalization of Pair 1
                Add(1);                         //Addition of Pair 2
                Align(2);                       //Alignment of Pair 3
                Right_Shift(3);                 //Right Shifting of Pair 4
                Compare(4);                     //Comparison of Pair 5
                printf("\nEND OF STAGE 4");
                break;
        }
        case 5:
        {
```

```
                        Normalize(1);           //Normalization of Pair 2
                        Add(2);                 //Addition of Pair 3
                        Align(3);               //Alignment of Pair 4
                        Right_Shift(4);         //Right Shifting of Pair 5
                        Compare(5);             //Comparison of Pair 6
                        printf("\nEND OF STAGE 5");
                        break;
                }
                case 6:
                {
                        Normalize(2);           //Normalization of Pair 3
                        Add(3);                 //Addition of Pair 4
                        Align(4);               //Alignment of Pair 5
                        Right_Shift(5);         //Right Shifting of Pair 6
                        printf("\nEND OF STAGE 6");
                        break;
                }
                case 7:
                {
                        Normalize(3);           //Normalization of Pair 4
                        Add(4);                 //Addition of Pair 5
                        Align(5);               //Alignment of Pair 6
                        printf("\nEND OF STAGE 7");
                        break;
                }
                case 8:
                {
                        Normalize(4);           //Normalization of Pair 5
                        Add(5);                 //Addition of Pair 6
                        printf("\nEND OF STAGE 8");
                        break;
                }
                case 9:
                {
                        Normalize(5);           //Normalization of Pair 6
                        printf("\nEND OF STAGE 9");
                        break;
                }
        }
        getch();
    }
}
void Compare(int i)
{
```

```
	printf("\n------------------------------Comparison of Exponent's-------------------------------
");

	//Loop to determine the shift and the greater number
	if(A_exp_18[i]>B_exp_18[i])
	{
		shift=A_exp_18[i]-B_exp_18[i];
	}
	else if(A_exp_18[i]<B_exp_18[i])
	{
		shift=B_exp_18[i]-A_exp_18[i];
	}
	else
	{
		shift=0;
	}

	//Including the 1 from "1.m" into the Mantissa
	mask1=0x400000;
	printf("\nTestbench %d",i+1);
	A_mantissa_18[i]=A_mantissa_18[i]>>1;
	B_mantissa_18[i]=B_mantissa_18[i]>>1;
	A_mantissa_18[i]=A_mantissa_18[i]|mask1;
	B_mantissa_18[i]=B_mantissa_18[i]|mask1;
	A_exp_18[i]=A_exp_18[i]+1;
	B_exp_18[i]=B_exp_18[i]+1;
}

void Right_Shift(int i)
{
printf("\n---------------------------Right Shifting of Mantissa's-----------------------------");

//Aligning the Mantissa based on the number of Right shifts required and making the
exponents same
		printf("\nTestbench %d",i+1);
		if(A_exp_18[i]<B_exp_18[i])
		{
			A_mantissa_18[i]=A_mantissa_18[i]>>shift;
			A_exp_18[i]=A_exp_18[i]+shift;
			Sum_sign_18[i]=B_sign_18[i];
		}
		else if(A_exp_18[i]>B_exp_18[i])
		{
			B_mantissa_18[i]=B_mantissa_18[i]>>shift;
```

```
                B_exp_18[i]=B_exp_18[i]+shift;
                Sum_sign_18[i]=A_sign_18[i];
        }
        else
        {
                A_mantissa_18[i]=A_mantissa_18[i];
                B_mantissa_18[i]=B_mantissa_18[i];
                Sum_sign_18[i]=A_sign_18[i];
        }
        Sum_exp_18[i]=A_exp_18[i];
}
void Align(int i)
{
        printf("\n----------Alignment----------");

        //Determining the smaller mantissa and taking it's 2's complement
        printf("\nTestbench %d",i+1);
        if(A_sign_18[i]!=B_sign_18[i])
        {
                if(A_mantissa_18[i]<B_mantissa_18[i])
                {
                        A_mantissa_18[i]=(~A_mantissa_18[i])+1;
                }
                else
                {
                        B_mantissa_18[i]=(~B_mantissa_18[i])+1;
                }
        }
        else
        {
                A_mantissa_18[i]=A_mantissa_18[i];
                B_mantissa_18[i]=B_mantissa_18[i];
        }
}

void Add(int i)
{
        printf("\n----------Addition----------");

        //Addition of the Mantissa's
        printf("\nTestbench %d",i+1);
        Sum_mantissa_18[i]=A_mantissa_18[i]+B_mantissa_18[i];
        Sum_exp_18[i]=A_exp_18[i];
}
```

```
void Normalize(int i)
{
        printf("\n----------------------------------Normalization----------------------------------");

        //Normalization of the Mantissa's
        printf("\nTestbench %d",i+1);
        mask2=0x7fffff;
                if(A_sign_18[i]!=B_sign_18[i])                  //For Sign(A) != Sign(B)
                {
                        if(A_sign_18[i]==0&B_sign_18[i]==1)     //For A=+ve & B=-ve
                        {
                                if(A_mantissa_18[i]==0)         //For A=0
                                {
                                        Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
                                         x[i]=Sum_mantissa_18[i]&mask2;
                                         x[i]=x[i]>>1;
                                         Sum_exp_18[i]=Sum_exp_18[i]-1;
        printf("\nNormalized Value of Sum is %d %x %lx\n" , Sum_sign_18[i], Sum_exp_18[i],
x[i]);
                                }
                                else                            //For A !=0
                                {
                                        Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
                                        x[i]=Sum_mantissa_18[i]&mask2;
                                        Sum_exp_18[i]=Sum_exp_18[i]-2;
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i],
Sum_exp_18[i],x[i]);
                                }
                        }

                        else                                    //For Sign(A) != Sign(B)
                        {
                                Sum_mantissa_18[i]=Sum_mantissa_18[i]<<2;
                                x[i]=Sum_mantissa_18[i]&mask2;
                                Sum_exp_18[i]=Sum_exp_18[i]-2;
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i], Sum_exp_18[i]
,x[i]);
                        }
                }
                else                                            //For Sign(A) = Sign(B)
                {
                        if(A_sign_18[i]==0&B_sign_18[i]==0)     //Both A & B = +ve
                        {
```

```
                        if(A_mantissa_18[i]==0)                 //For A=0
                        {
                                x[i]=Sum_mantissa_18[i]&mask2;
                                Sum_exp_18[i]=Sum_exp_18[i]-1;
                        }

        else                                    //For A != 0
                        {
                                x[i]=Sum_mantissa_18[i]&mask2;
                        }
                }
                else if(A_sign_18[i]==1&B_sign_18[i]==1)    //Both A & B = -ve
                {
                        Sum_mantissa_18[i]=Sum_mantissa_18[i]<<1;
                        Sum_exp_18[i]=Sum_exp_18[i]-1;
                        x[i]=Sum_mantissa_18[i]&mask2;
                }
            else{}
        printf("\nNormalized Value of Sum is %d %x %lx\n", Sum_sign_18[i], Sum_exp_18[i],
x[i]);
            }
}
```

OUTPUT:

STAGE – 0

```
------------------------------Comparison of Exponent's--------------------------------
Testbench 1
END OF STAGE 0
```

STAGE – 1

```
---------------------------Right Shifting of Mantissa's-------------------------------
Testbench 1
------------------------------Comparison of Exponent's--------------------------------
Testbench 2
END OF STAGE 1
```

STAGE – 2

```
-------------------------------------Alignment--------------------------------------
Testbench 1
---------------------------Right Shifting of Mantissa's-------------------------------
Testbench 2
------------------------------Comparison of Exponent's--------------------------------
Testbench 3
END OF STAGE 2
```

STAGE – 3

```
-----------------------------------Addition----------------------------------------
Testbench 1
-----------------------------------Alignment---------------------------------------
Testbench 2
---------------------------Right Shifting of Mantissa's-----------------------------
Testbench 3
-----------------------------Comparison of Exponent's-------------------------------
Testbench 4
END OF STAGE 3
```

STAGE – 4

```
-----------------------------------Normalization-----------------------------------
Testbench 1
Normalized Value of Sum is 0 87 a8000

-----------------------------------Addition----------------------------------------
Testbench 2
-----------------------------------Alignment---------------------------------------
Testbench 3
---------------------------Right Shifting of Mantissa's-----------------------------
Testbench 4
-----------------------------Comparison of Exponent's-------------------------------
Testbench 5
END OF STAGE 4
```

STAGE – 5

```
-----------------------------------Normalization-----------------------------------
Testbench 2
Normalized Value of Sum is 0 84 480000

-----------------------------------Addition----------------------------------------
Testbench 3
-----------------------------------Alignment---------------------------------------
Testbench 4
---------------------------Right Shifting of Mantissa's-----------------------------
Testbench 5
-----------------------------Comparison of Exponent's-------------------------------
Testbench 6
END OF STAGE 5
```

STAGE – 6

```
-----------------------------------Normalization-----------------------------------
Testbench 3
Normalized Value of Sum is 0 84 500000

-----------------------------------Addition----------------------------------------
Testbench 4
-----------------------------------Alignment---------------------------------------
Testbench 5
---------------------------Right Shifting of Mantissa's-----------------------------
Testbench 6
END OF STAGE 6
```

STAGE – 7

```
-----------------------------------Normalization-----------------------------------
Testbench 4
Normalized Value of Sum is 1 87 2e8000

-------------------------------------Addition-------------------------------------
Testbench 5
-------------------------------------Alignment------------------------------------
Testbench 6
END OF STAGE 7
```

STAGE – 8

```
-----------------------------------Normalization-----------------------------------
Testbench 5
Normalized Value of Sum is 0 1 0

-------------------------------------Addition-------------------------------------
Testbench 6
END OF STAGE 8
```

STAGE – 9

```
-----------------------------------Normalization-----------------------------------
Testbench 6
Normalized Value of Sum is 1 84 3c0000

END OF STAGE 9
```