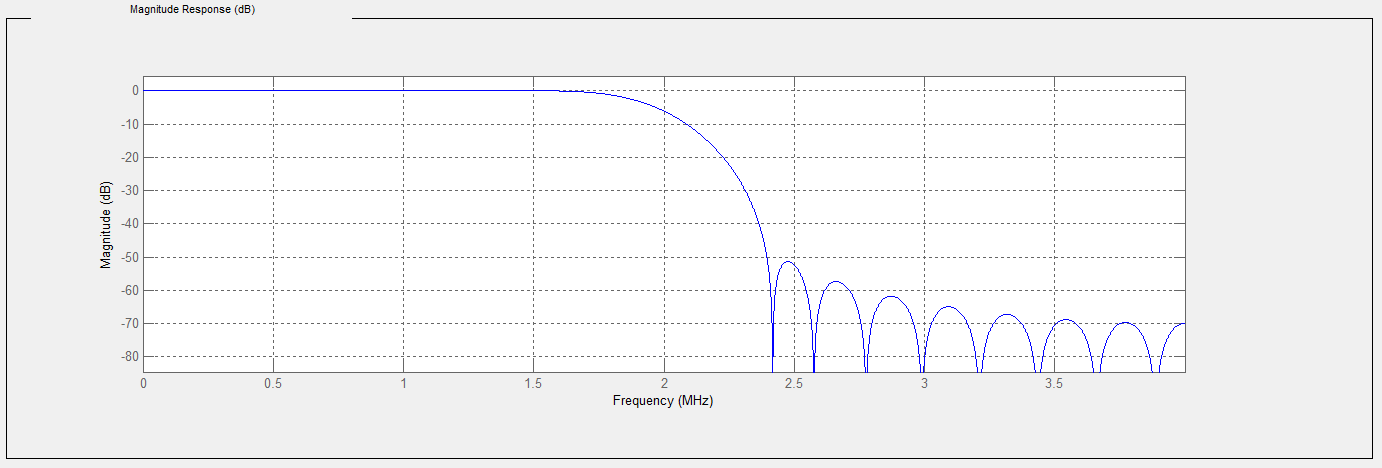
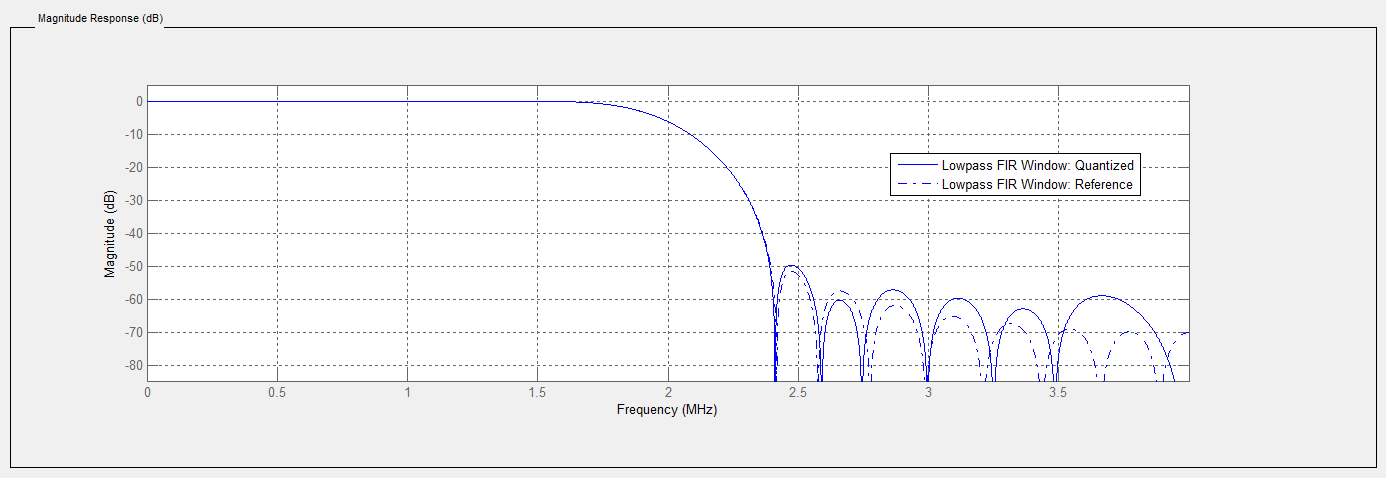
Experiment I：FIR Filter Derign

**I.Curve of filter’s frequency response**

1.curve of the frequency response

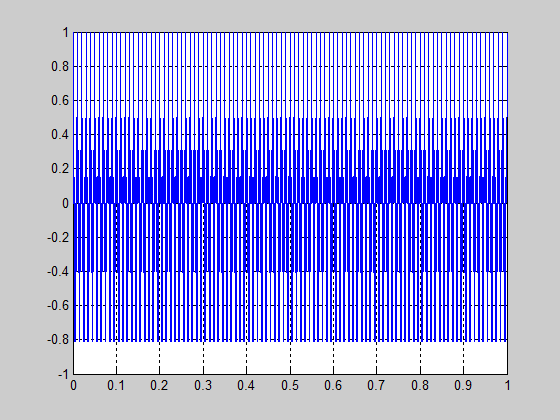


Gpraph 1: frequency response of float-coefficient filter



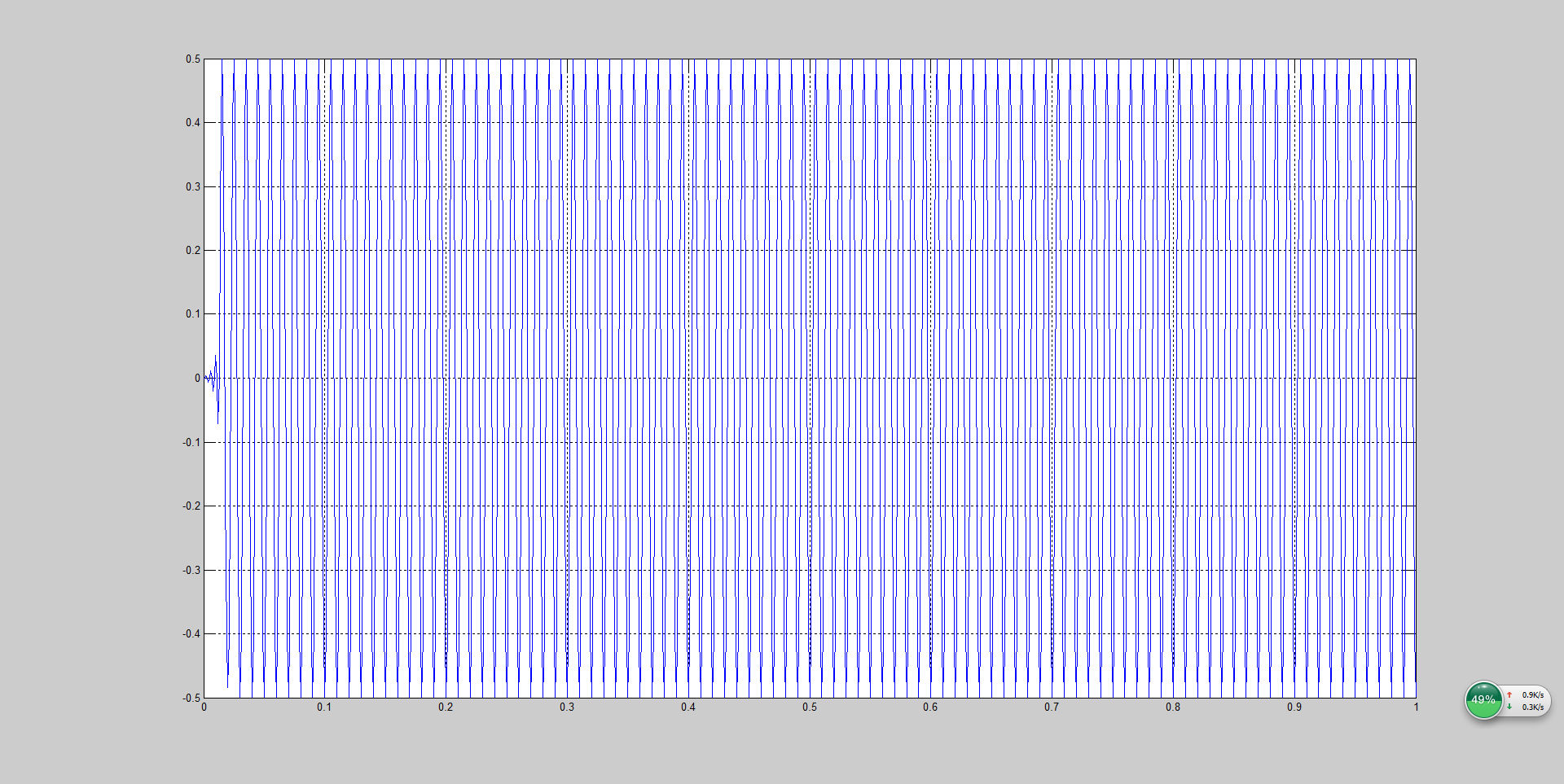
Graph 2:curve in dot is the frequency response of fixed-coefficient filter

2. curve of the input signal

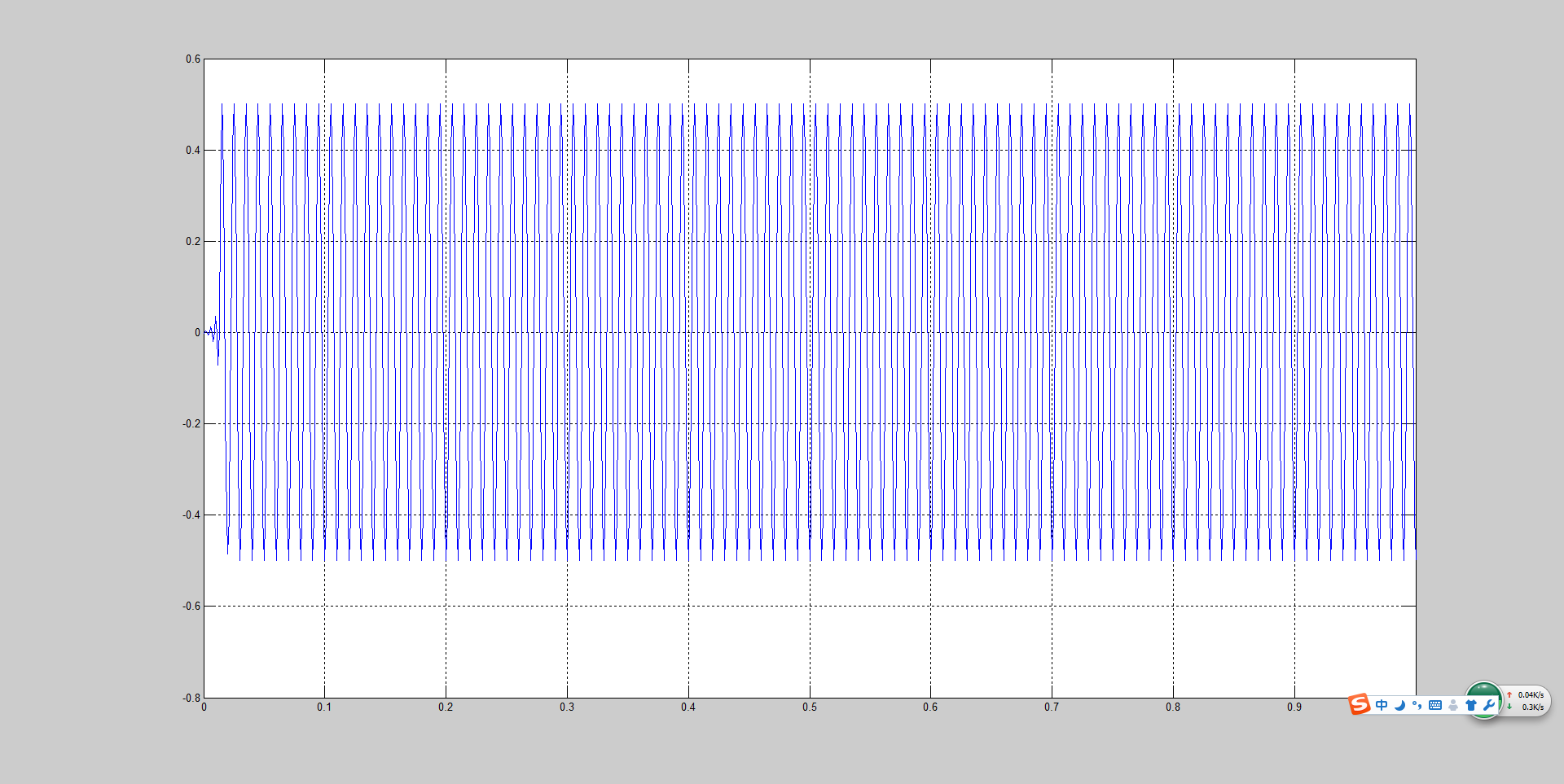


Graph 3: input=(cos(2\*pi\*100\*t)+cos(2\*pi\*400\*t))/2. The graph is drawn in matlab

3. curve of output with filter



Graph 4: curve of output of fixed-coefficient filter drawn in the matlab



Graph 5: curve of output of the float-coefficient filter drawn in the matlab

**II.Filter type, fixed width, and order of the filter**

**1.filter coefficients**

The filter is Kaiser-window 30 order filter. The width of the fixed coefficient is 12bit. The width of the input data is 16bits.

The tables below shows the coefficients of the filter’s taps.

Table 1: coefficients of fixed filter

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1-5 | 0.0009765625 | 0 | 0.00390625 | 0 | 0.00830078125 |
| 6-10 | 0 | 0.01611328125 | 0 | 0.02880859375 | 0 |
| 11-15 | 0.05078125 | 0 | 0.09814453125 | 0 | 0.3154296875 |
| 16-20 | 0.5 | 0.3154296875 | 0 | 0.09814453125 | 0 |
| 21-25 | 0.05078125 | 0 | 0.02880859375 | 0 | 0.01611328125 |
| 26-30 | 0 | 0.00830078125 | 0 | 0.00390625 | 0 |
| 31 | 0.0009765625 |  |  |  |  |

Table 2: coefficients of floated filter

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1-5 | 0.00117897878546426 | 1.92785427471943e-18 | 0.00374191492448100 | 4.22630133562692e-18 | 0.00838910874245180 |
| 6-10 | 7.21986825734144e-18 | 0.0161363511903562 | 1.06183656444815e-17 | 0.0287904884886270 | 1.39980253423359e-17 |
| 11-15 | 0.0507249870156716 | 1.68775579034484e-17 | 0.0979179687048844 | 1.88142223102893e-17 | 0.315593832728724 |
| 16-20 | 0.500158917724390 | 0.315593832728724 | 1.88142223102893e-17 | 0.0979179687048844 | -1.68775579034484e-17 |
| 21-25 | 0.0507249870156716 | 1.39980253423359e-17 | 0.0287904884886270 | 1.06183656444815e-17 | 0.0161363511903562 |
| 26-30 | 7.21986825734144e-18 | 0.00838910874245180 | 4.22630133562692e-18 | 0.00374191492448100 | 1.92785427471943e-18 |
| 31 | 0.00117897878546426 |  |  |  |  |

**2. Error analysis**

**1)error analysis of float coefficient filter.(float input)**

Code:

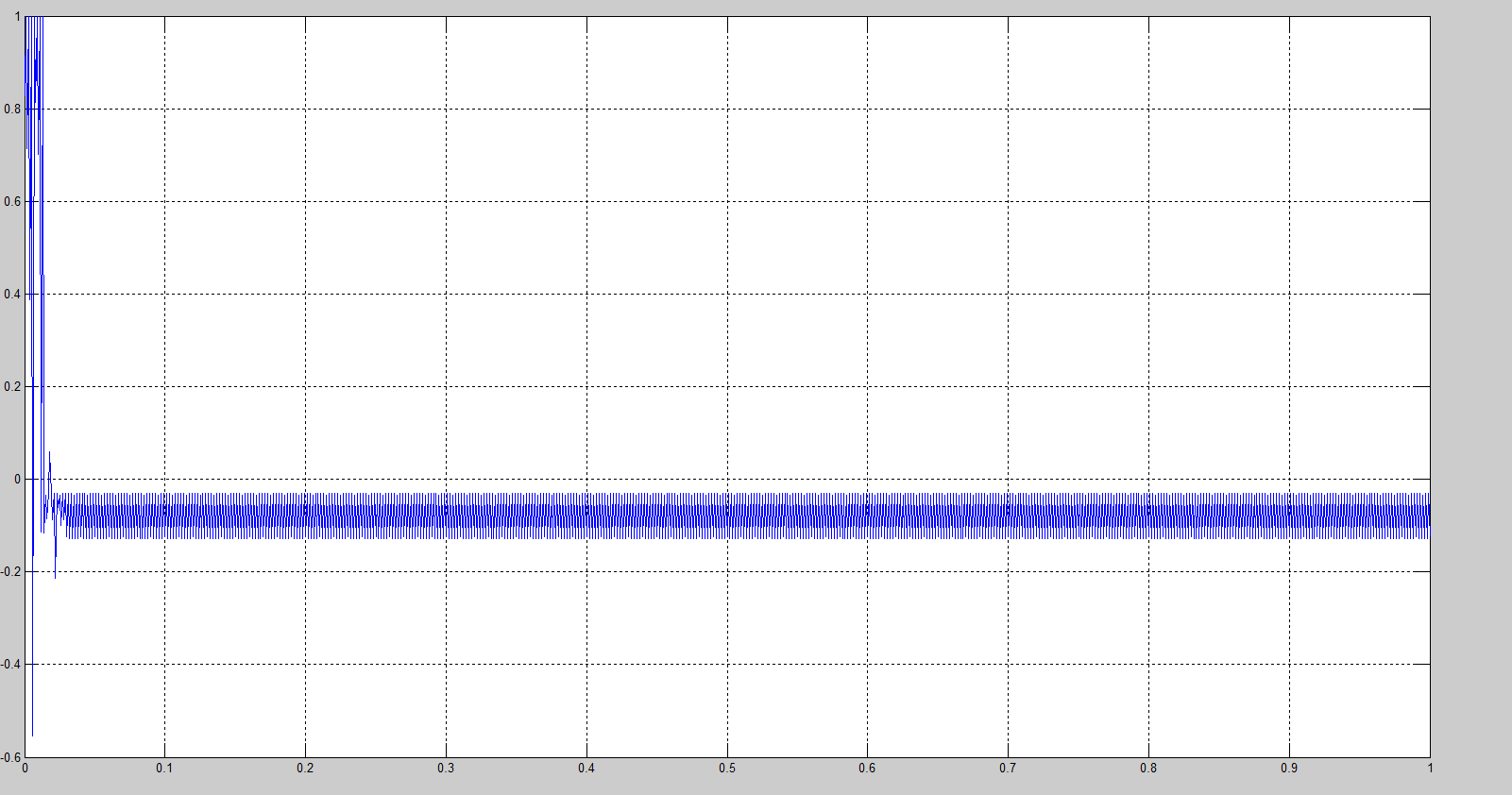
input=(cos(2\*pi\*100\*t)+cos(2\*pi\*400\*t))/2;

>> yflo=filter(flo\_filter,input); %float coefficient, float input

>> yfix=filter(fix\_filter,input); %fixed coefficient, float input

yer=100\*(yfix-yflo)./yflo;

>> plot(t,yer);grid on;



2)

input=(cos(2\*pi\*100\*t)+cos(2\*pi\*400\*t))/2;

>> yflo=filter(flo\_filter,input); %float coefficient, float input

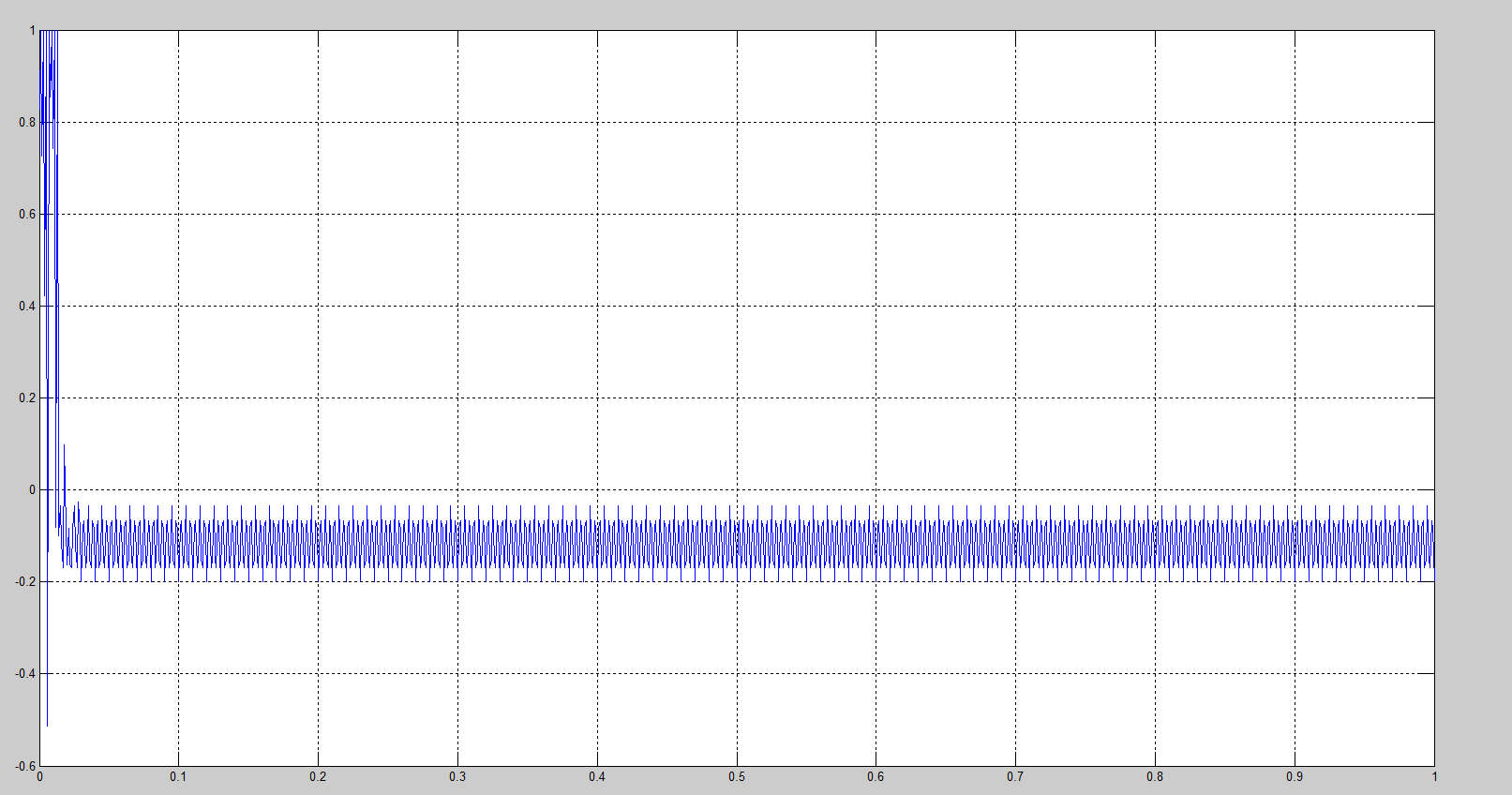
input\_12=fi(input,1,12);

yfi=filter(fix\_filter,input\_12); %fixed coefficient, fixed input

>> figure;

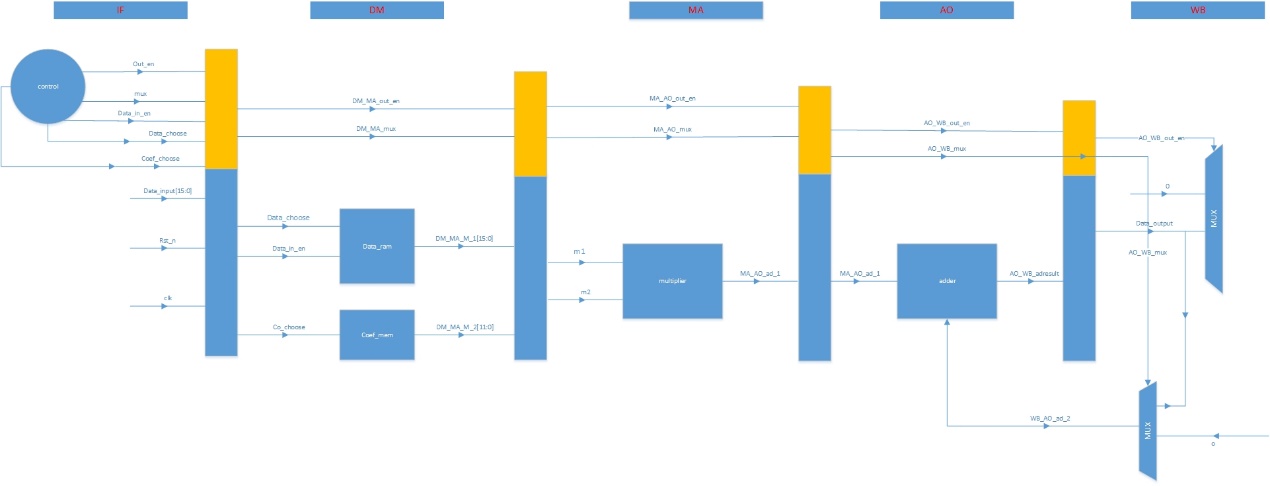
>> yerr=100\*(yfi-yflo)./yflo;

>> plot(t,yerr);grid on;



The error is about -0.2%.

**III hardware structure of the filter**

1.The structure is cascaded into 5 stages. IF->DM->MA->AO->WB.

**2.main modules and signal**

**1)IF stage:main module is control module**

Input:

clk, rst\_n，data\_in[15:0]

Output:

IF\_DM\_ut\_en:used for the output data control

IF\_DM\_mux: used for the adder. Choose 0 or adresult to add.

IF\_DM\_data\_in\_en: control whether data can be written into data memory

IF\_DM\_data\_in:the input data of the filter

data\_choose: choose data from data ram

co\_choose:choose the coefficient for multiplier

Others are the internal registers of IF stage

**2)DM stage:contains two main module data\_ram and coefficient\_rom**

Input(the output of the IF stage):

clk,rst\_n

data\_choose

co\_choose

IF\_DM\_data\_in[15:0]

IF\_DM\_mux,

IF\_DM\_data\_in\_en,

IF\_DM\_out\_en

Output(input of the next stage):

DM\_MA\_M\_1[15:0]:to be the one of the two multipliers.

DM\_MA\_M\_2[11:0]:to be the other one of the two multipliers.

DM\_MA\_out\_en

DM\_MA\_mux

**3)MA stage:only deals with the multiply operation**

Input:

clk,rst\_n,

DM\_MA\_M\_1

DM\_MA\_M\_2

DM\_MA\_out\_en

DM\_MA\_mux

Output:

MA\_AO\_ad\_1[26:0]

MA\_AO\_out\_en,

MA\_AO\_mux

4)**AO stage:only deals with the add operation**

Input:

MA\_AO\_ad\_1[26:0]:first add number

WB\_A0\_ad\_2[27:0]:second add number, back from WB stage

MA\_AO\_out\_en

MA\_AO\_mux

Output:

AO\_WB\_adresult[27：0]

AO\_WB\_out\_en

AO\_WB\_mux

5)WB stage:only decide whether to give out the data or to adder

Input:

AO\_WB\_adresult[27：0]

AO\_WB\_out\_en

AO\_WB\_mux

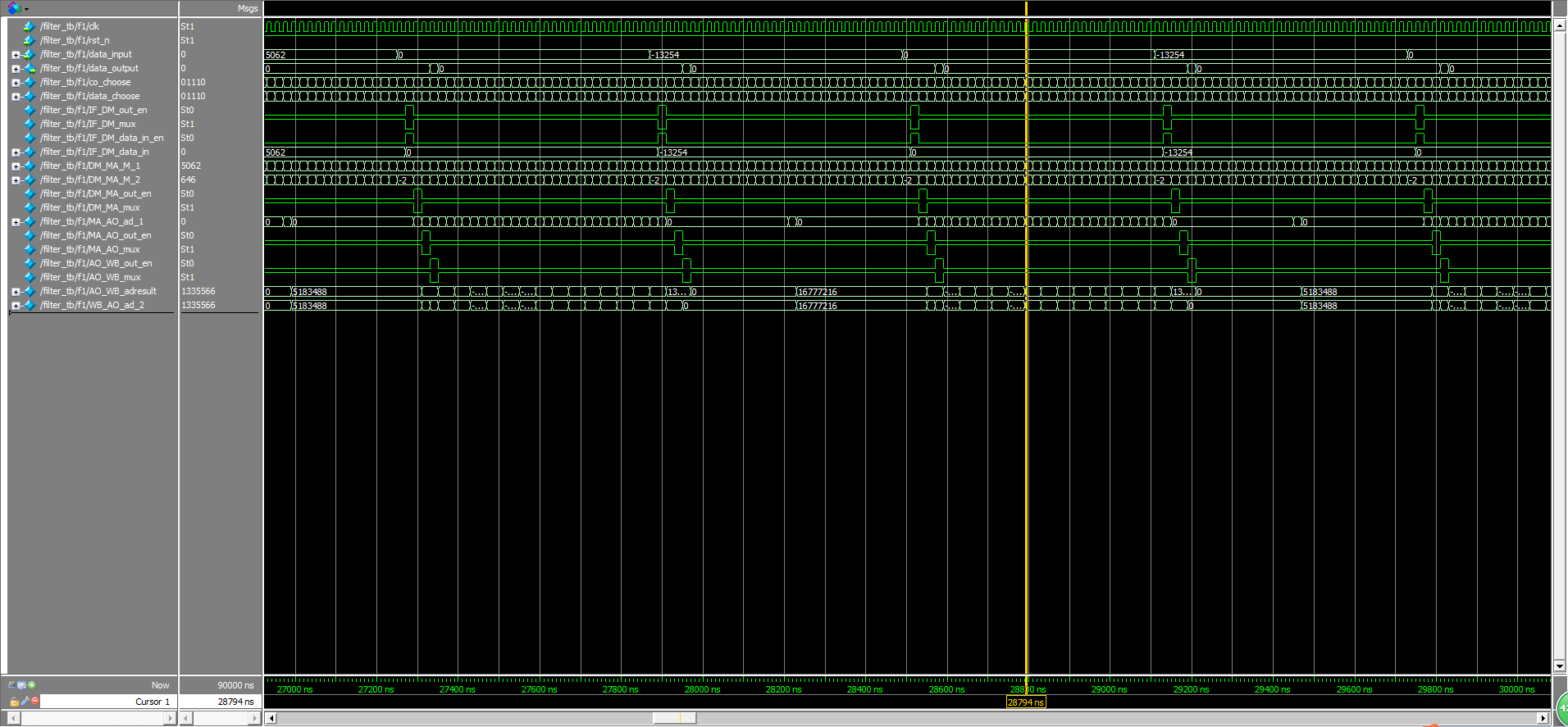
Output:

data\_out[27:0]

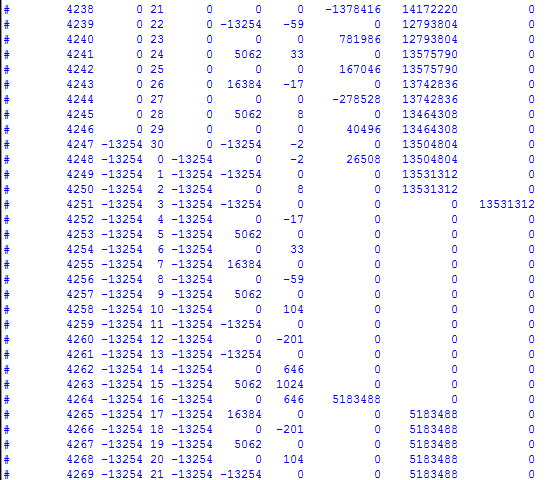
WB\_AO\_ad\_2[a7:0]

**IV RTL simulation results and data path and LTU area.**

1. modelsim simulation



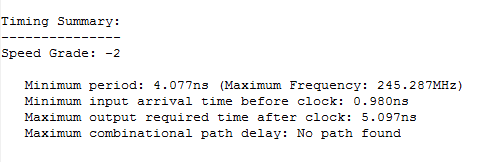
Waveforms of the simulation



data displayed during the simulation

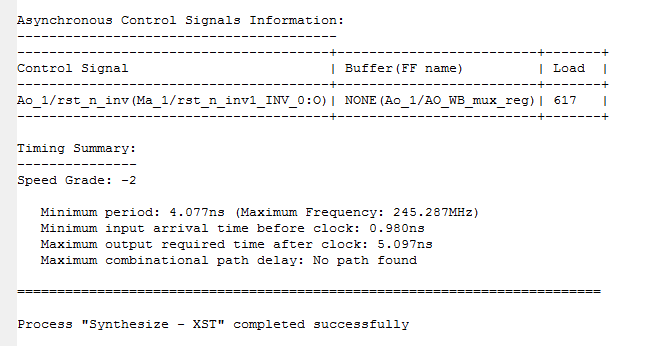
2.ISE synthesis results

1). The frequency is 245.287MHZ



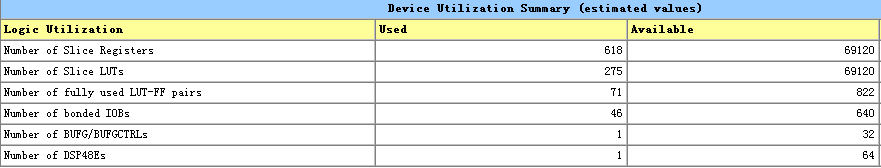
Graph from ISE

2)citical path: 4.077ns



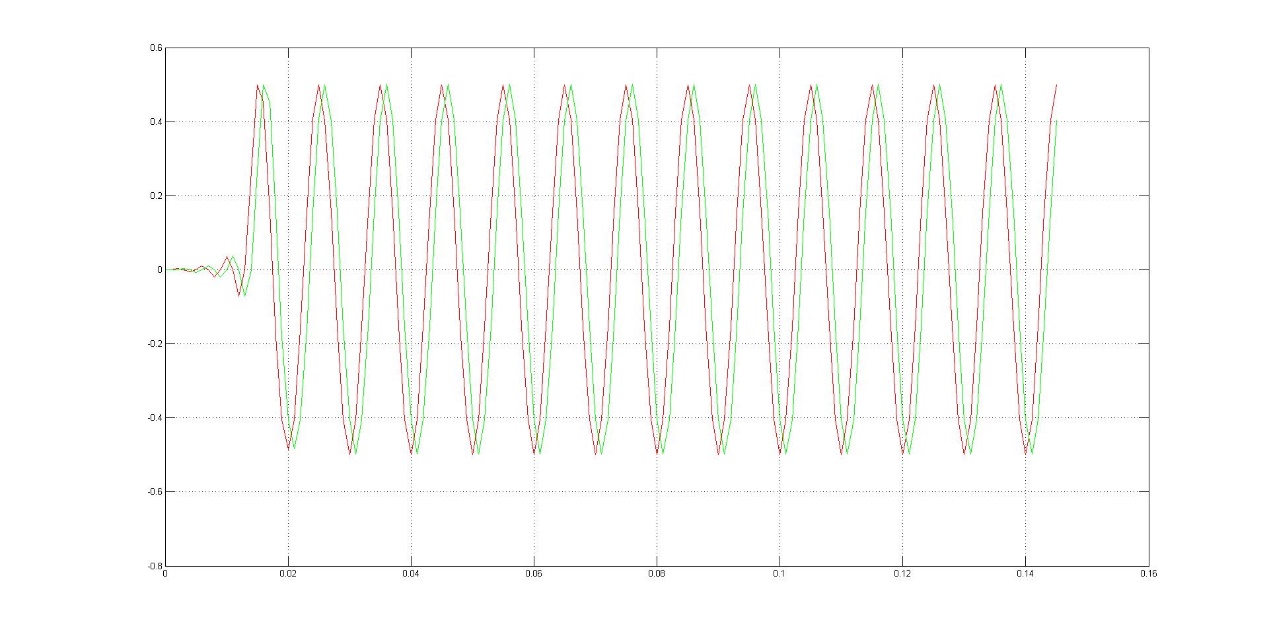
Graph from ISE

3)Area dissipation:



Graph from ISE

3)Final result of the filter



Final comparative graph of the output

**V Relative questions or materials**

1)

To get the right input, I find a matlab program to transform the floated input data. The code is as bellow:

function [numbin] = dec2binPN(numdec,N)

if (numdec>= 0)

numbin1 = dec2bin(numdec,N);

else

numbin1 = dec2bin(abs(numdec),N);

l1=length(numbin1);

numbin4=0;

for i=1:l1

if (numbin1(l1-i+1)==num2str(1))

numbin4=numbin4+0;

else

numbin4=numbin4+2^(i-1);

end

end

numbin4=numbin4+1;

numbin5=dec2bin(numbin4);

numbin1=num2str(numbin5,N);

end

numbin=numbin1;

end

(ps:this is really difficult for me because I don’t know how to use matlab at first. But this really improves my personal ability of dealing with obstacles that I’ve never met)

**VI Conclusions and Rethink of the Lab**

1)

I didn’t understand the rule of the signed binary number calculation. So I converted the number wrongly. This resulted to my unable to find the real error of my design. And this also caused me always finding errors of the design logic.

Signed [11:0]←floated number\*2^11; at first I multiplied 2^12

Signed [15:0]←floated number\*2^15; at first I multiplied 2^16

Both are wrong. So I couldn’t get the rignt waveforms all the time. But this teaches me that study and enginner needs 100% care and caution. To save time, each step must be right.

2)

At first, the time logic of the circuit is not clear for me. So I changed the structure of the FIR.

3)

I practiced my skill of coding and get more familiar with the tool of the modelsim. $fmonitor and $fdisplay is totally different. And I tried a lot of method to find out my errors. This process made me learned a lot.

4)

I also used C++ to deal with the problems of dealing with files and binary data’s width. Bellow is the code of one program which is desigend to fix all the input data to 16 bit width:

#include<iostream>

#include<fstream>

using namespace std;

int main()

{

char tmp[50];

char tmp1[50], tmp2[50];

int count = 0;

int len = 0;

ifstream infile;

ifstream infile\_2;

ofstream oufile;

infile.open("inputfix.txt");

oufile.open("inputfix\_n.txt");

if (!infile)

{

cout << "\nCan't open file";

}

if (!oufile)

{

cout << "\nCan't open file";

}

while (!infile.eof())

{

infile >> tmp;

len = strlen(tmp);

if (len < 16)

{

for (int i = len; i <=15; i++)

{

tmp[i] = '0';

}

}

if (len>16)

{

for (int i = len-1; i>15; i--)

{

tmp[i] = '\0';

}

}

oufile << tmp << '\n';

cout << tmp << '\n';

}

}

5)

I think communicating with teacher and partner is very important. I didi this alone, and this takes me much time.