PANIMALAR ENGINEERING COLLEGE

(A CHRISTIAN MINORITY INSTITUTION)

JAISAKTHI EDUCATIONAL TRUST

ACCREDITED BY NATIONAL BOARD OF ACCREDITATION

BANGALORE TRUNK ROAD, VARADHARAJAPURAM,

NASARATHPET, POONAMALLEE,

CHENNAI - 600 123.

DEPARTMENT OF

ELECTRONICS & COMMUNICATION ENGINEERING

EC6511 DIGITAL SIGNAL PROCESSING LAB

LAB MANUAL

V SEMESTER ECE

(2017 - 2018 ODD SEMESTER)

DEPARTMENT OF ECE

VISION

To emerge as a centre of excellence in providing quality education and produce technically competent Electronics and Communication Engineers to meet the needs of industry and Society.

MISSION

M1: To provide best facilities, infrastructure and environment to its students, researchers and faculty members to meet the Challenges of Electronics and Communication Engineering field.

M2: To provide quality education through effective teaching – learning process for their future career, viz placement and higher education.

M3: To expose strong insight in the core domains with industry interaction.

M4: Prepare graduates adaptable to the changing requirements of the society through life long learning.

PROGRAMME EDUCATIONAL OBJECTIVES

1. To prepare graduates to analyze, design and implement electronic circuits and systems using the knowledge acquired from basic science and mathematics.
2. To train students with good scientific and engineering breadth so as to comprehend, analyze, design and create novel products and solutions for real life problems.
3. To introduce the research world to the graduates so that they feel motivated for higher studies and innovation not only in their own domain but multidisciplinary domain.
4. Prepare graduates to exhibit professionalism, ethical attitude, communication skills, teamwork and leadership qualities in their profession and adapt to current trends by engaging in lifelong learning.
5. To practice professionally in a collaborative, team oriented manner that embraces the multicultural environment of today's business world.

PROGRAMME OUTCOMES

1. **Engineering Knowledge:** Able to apply the knowledge of Mathematics, Science, Engineering fundamentals and an Engineering specialization to the solution of complex Engineering problems.
2. **Problem Analysis:** Able to identify, formulate, review research literature, and analyze complex Engineering problems reaching substantiated conclusions using first principles of Mathematics, Natural sciences, and Engineering sciences.
3. **Design / Development of solutions:** Able to design solution for complex Engineering problems and design system components or processes that meet the specified needs with appropriate considerations for the public health and safety and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Able to use Research - based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Able to create, select and apply appropriate techniques, resources, and modern Engineering IT tools including prediction and modeling to complex Engineering activities with an understanding of the limitations.
6. **The Engineer and society: A**ble to apply reasoning informed by the contextual knowledge to access societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional Engineering practice.
7. **Environment and sustainability:** Able to understand the impact of the professional Engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Able to apply ethical principles and commit to professional ethics and responsibilities and norms of the Engineering practice.
9. **Individual and Team work:** Able to function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Able to communicate effectively on complex Engineering activities with the Engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project Management and Finance:** Able to demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life – long learning:** Able to recognize the needs for, and have the preparation and ability to engage in independent and life-long learning in the broadest contest of technological change.

PROGRAMME SPECIFIC OUTCOMES

1. Graduates should demonstrate an understanding of the basic concepts in the primary area of Electronics and Communication Engineering, including: analysis of circuits containing both active and passive components, electronic systems, control systems, electromagnetic systems, digital systems, computer applications and communications.
2. Graduates should demonstrate the ability to utilize the mathematics and the fundamental knowledge of Electronics and Communication Engineering to design complex systems which may contain both software and hardware components to meet the desired needs.
3. The graduates should be capable of excelling in Electronics and Communication Engineering industry/Academic/Software companies through professional careers.

ANNA UNIVERSITY, CHENNAI
R – 2013 SYLLABUS

EC6511 DIGITAL SIGNAL PROCESSING LABORATORY

OBJECTIVES
The student should be made to

- To implement Linear and Circular Convolution
- To implement FIR and IIR filters
- To study the architecture of DSP processor
- To demonstrate Finite word length effect

LIST OF EXPERIMENTS:
MATLAB / EQUIVALENT SOFTWARE PACKAGE

- Generation of sequences (functional & random) & correlation
- Linear and Circular Convolutions
- Spectrum Analysis using DFT
- FIR filter design
- IIR filter design
- Multirate Filters
- Equalization

DSP PROCESSOR BASED IMPLEMENTATION

- Study of architecture of Digital Signal Processor
- MAC operation using various addressing modes
- Linear Convolution
- Circular Convolution
- FFT Implementation
- Waveform generation
- IIR and FIR Implementation
- Finite Word Length Effect

LIST OF EXPERIMENTS

MATLAB PROGRAMS

1. Generation of sequences (functional & random) Unit Impulse Sequence
   - Unit Step Sequence
   - Ramp Sequence
   - Exponential Sequence
   - Sine Sequence
   - Cosine Sequence
2. Correlation
3. Linear and Circular Convolutions
4. Spectrum Analysis using DFT
5. FIR Filter Design
   - Rectangular Window
   - Blackman Window
   - Hamming Window
   - Hanning Window
6. IIR Filter Design
7. Multirate Filters
8. Equalization

PROCESSOR PROGRAMS (TMS 320 C5416 DSK)

1. Study of architecture of Digital Signal Processor
2. Linear Convolution
3. Circular Convolution
4. FFT Implementation
5. Waveform Generation
   - Sine Wave
   - Cos Wave
6. IIR and FIR Implementation

COURSE OUTCOMES:

At the end of the course, the student will be able to:

CO 1: Develop and experiment coding from basic mathematical operations to complex operations like DFT and FFT.
CO 2: Visualize the amplitude and phase spectrum of the signal in frequency domain.
CO 3: Simulate FIR and IIR filter using MATLAB and DSP processor.
CO 4: Analyze the finite word length effect.

SYLLABI OF VARIOUS PREMIER INSTITUTIONS

IIT- KHARAGPUR

DIGITAL SIGNAL PROCESSING VIRTUAL LAB:

1. Study of Sampling theorem, effect of undersampling.
2. Study of Quantization of continuous – amplitude, discrete- time analog signals.
3. Study of different types of Companding Techniques.
4. Study of properties of Linear Time- Invariant system.
5. Study of Convolution: Series and Parallel system.
6. Study of Discrete Fourier Transform (DFT) and its inverse.
7. Study of Transform domain properties and its use
8. Study of FIR filter design using window method: Lowpass and highpass filter.
9. Study of FIR filter design using window method: Bandpass and Bandstop filter.
10. Study of Infinite Impulse Response (IIR) filter.

BRIDGING THE CURRICULUM GAP

To have more exposure in **Digital Signal Processing**, other than the experiments given in the syllabus of Anna University, experiment such as **"Study of sampling theorem, effect of undersampling"** and **"Study of properties of Linear Time- Invariant system"** as included as Content beyond Syllabus from **IIT- KHARAGPUR**.

- **Course Instructors**

CYCLE OF EXPERIMENTS

I CYCLE

1. Generation of sequences (functional & random) **(Using Matlab)**
   - Unit Impulse Sequence
   - Unit Step Sequence
   - Ramp Sequence
   - Exponential Sequence
   - Sine Sequence
   - Cosine Sequence
2. Correlation **(Using Matlab)**
3. Linear and Circular Convolution **(Using Matlab)**
4. Spectrum Analysis using DFT **(Using Matlab)**
5. Study of architecture of Digital Signal Processor of TMS 320C5416
6. Linear Convolution **(Using TMS 320C5416 Processor)**
7. Circular Convolution **(Using TMS 320C5416 Processor)**
8. Waveform Generation **(Using TMS 320C5416 Processor)**
   - Sine Wave
   - Cos Wave

II CYCLE

1. FIR Filter Design **(Using Matlab)**
   - Rectangular Window
   - Blackman Window
   - Hamming Window
   - Hanning Window
2. IIR Filter Design **(Using Matlab)**
3. Multirate Filters **(Using Matlab)**
4. Equalization **(Using Matlab)**
5. IIR and FIR Implementation **(Using TMS 320C5416 Processor)**
6. FFT Implementation **(Using TMS 320C5416 Processor)**

ADDITIONAL EXPERIMENTS

1. Sampling & Effect of Aliasing **(Using Matlab)**
2. Calculation of FFT of a signal **(Using Matlab)**
3. Down Sampling and Up Sampling **(Using Matlab)**
4. Waveform Generation **(Using Matlab)**
   - Triangular Waveform
   - Saw Tooth Waveform
   - Square Waveform
5. Sum of sinusoidal signals. **(Using Matlab)**
6. Wave generation. **(Using TMS 320C5416 Processor)**

ADDITIONAL EXPERIMENTS

S.NO	NAME OF THE EXPERIMENTS
1	Sampling & Effect of Alaising **(using MATLAB)**
2	Calculation of FFT of a signal **(using MATLAB)**
3	Down Sampling and UpSampling **(using MATLAB)**
4	Wave form Generation- Triangular, Sawtooth and Square **(using MATLAB)**
5	Sum of sinusoidal signals **(using MATLAB)**
6	Wave generation **(using TMS320C5416 processor)**

CONTENT BEYOND SYLLABUS

S.NO	NAME OF THE EXPERIMENTS
1	Computation of power density spectrum of a sequence
2	Frequency response of anti imaging and anti aliasing filters
3	CD data to DVD data
4	Study of property of linear time invariant system

CONTENT

CYCLE- I

GENERATION OF SEQUENCES

EXPERIMENT NO: 1 DATE:

AIM:

To generate basic sequences such as Unit impulse, Unit step, Ramp, Exponential, Sine sequence & Cosine Sequence using MATLAB Programs.

PROGRAMS:

% PROGRAM FOR GENERATION OF UNIT IMPULSE SIGNAL

```
clc; clear all; close all;
t = -2 : 1: 2;
y = [zeros(1,2) ,ones(1,1) ,zeros(1,2)];
subplot(2,2,1); stem(t,y);
title('UNIT IMPULSE');ylabel('Amplitude---->') ;xlabel(' (a)n---->');
```

% PROGRAM FOR GENERATION OF UNIT STEP SEQUENCE [u(n)-u(n-N)

```
n = input ('Enter the N Value: ');
t= 0 : 1: n-1;
y1 = ones(1,n);
subplot (2,2,2); stem (t, y1) ;
title('UNIT STEP');ylabel( 'AMPLITUDE---->'); xlabel('(b)n---->');
```

% PROGRAM FOR GENERATION OF RAMP SEQUENCE

```
N1= input ('Enter the Length of Ramp Sequence: ');
t = 0 : N1;
subplot(2,2,3);stem(t,t);
title('RAMP SEQUENCE');
ylabel( 'AMPLITUDE--->'); xlabel(' (c)n----> ');
```

% PROGRAM FOR GENERATION OF EXPONENTIAL SEQUENCE

```
n2 = input(' Enter the Length of Exponential Sequence:');
t = 0 : n2;
a = input ('Enter the Value:');
y2 = exp(a*t);
subplot(2,2,4);stem(t,y2);
title('EXPONENTIAL SEQUENCE');
ylabel( 'AMPLITUDE----->'); xlabel(' (d)n-----> ');
```

% PROGRAM FOR GENERATION OF SINE SEQUENCE

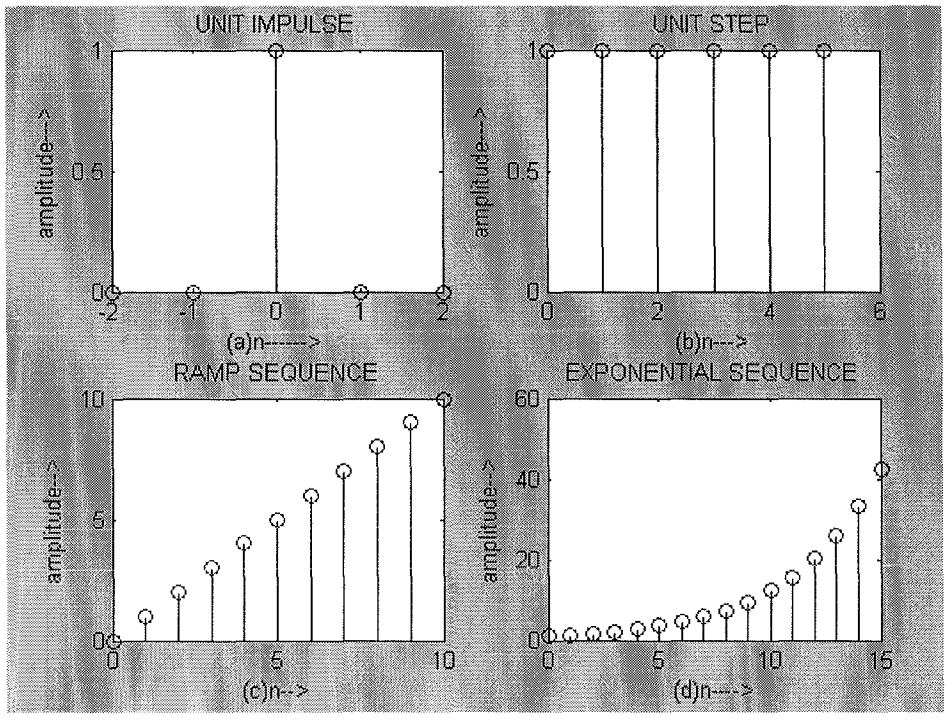
```
t = 0 :.01:pi;
y = sin(2 * pi* t);
figure (2) ;subplot(2,1,1);plot(t,y);
title('SINE WAVE');ylabcl( 'AMPLITUDE----->'); xlabel(' (a)t----->');
```

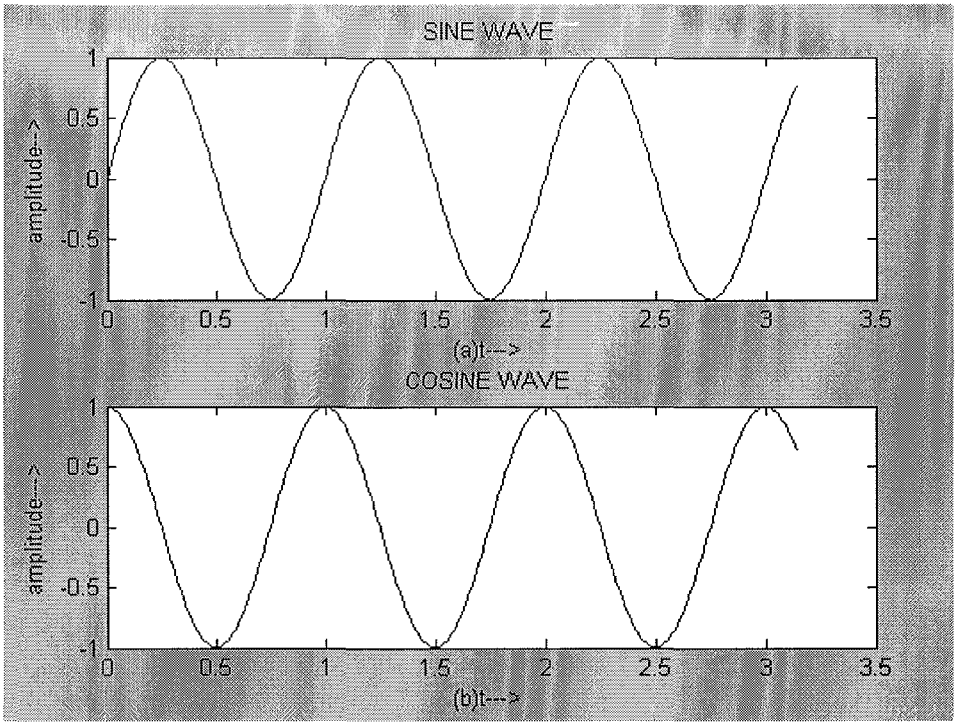
% PROGRAM FOR GENERATION OF COSINE SEQUENCE

```
t = 0 : .01:pi;
y = cos(2 * pi* t);
subplot(2,1,2);
plot(t,y);
title('COS WAVE');ylabel ( 'AMPLITUDE------>');
xlabel (' (b) t----->') ;
```

As an EXAMPLE:

Enter the N Value : **6**
Enter the Length of Ramp Sequence : **10**
Enter the Length of Exponential Sequence : **15**
Enter the Value :**0.25**





RESULT:

Thus the basic sequences of Unit impulse, Unit step, Ramp, Exponential and Sine Sequence & Cosine Sequence are generated using MATLAB Programs.

CORRELATION

EXPERIMENT NO: 2 DATE:

AIM:

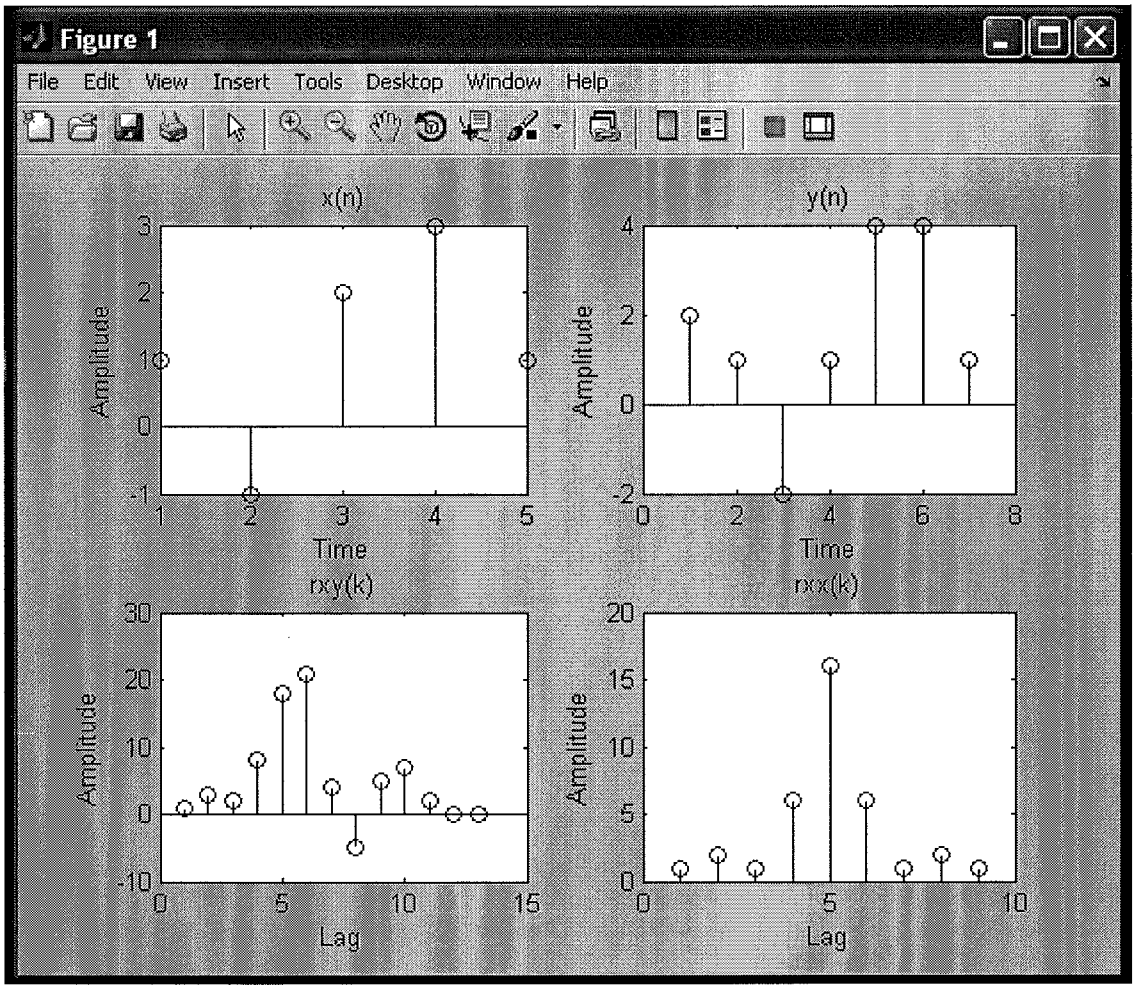
To implement auto-correlation and cross-correlation functions using MATLAB.

PROGRAM:

```
% Cross – Correlation and Auto – Correlation of finite sequences
clc;
clear all;
close all;
x = input ('Enter the Signal x =');
y = input ('Enter the Signal y =');
rxy = xcorr(x,y);   % cross-correlation of x and y
rxx = xcorr(x);     % Auto – Correlation of x
subplot(221);
stem(x);xlabel('Time');ylabel('Amplitude');title('x(n)');

subplot(222);
stem(y);
xlabel('Time');ylabel('Amplitude');title('y(n)');

subplot(223);
stem(rxy);
xlabel('Lag');ylabel('Amplitude');title('rxy(k)');

subplot(224);
stem(rxx);
xlabel('Lag');ylabel('Amplitude');title('rxx(k)');
```

As an EXAMPLE:

Enter the Signal x = [1 -1 2 3 1]
Enter the Signal y = [2 1 -2 1 4 4 1]



RESULT:

Thus the auto-correlation and cross-correlation functions are generated using MATLAB.

LINEAR AND CIRCULAR CONVOLUTION

EXPERIMENT NO: 3 DATE:

AIM:

To perform Convolution of two discrete sequences using
a) Linear Convolution.
b) Circular Convolution.

FORMULA:

$$y(n) = \sum_{k=-\infty}^{\infty} [x(k)h(n-k)]$$

PROGRAM:

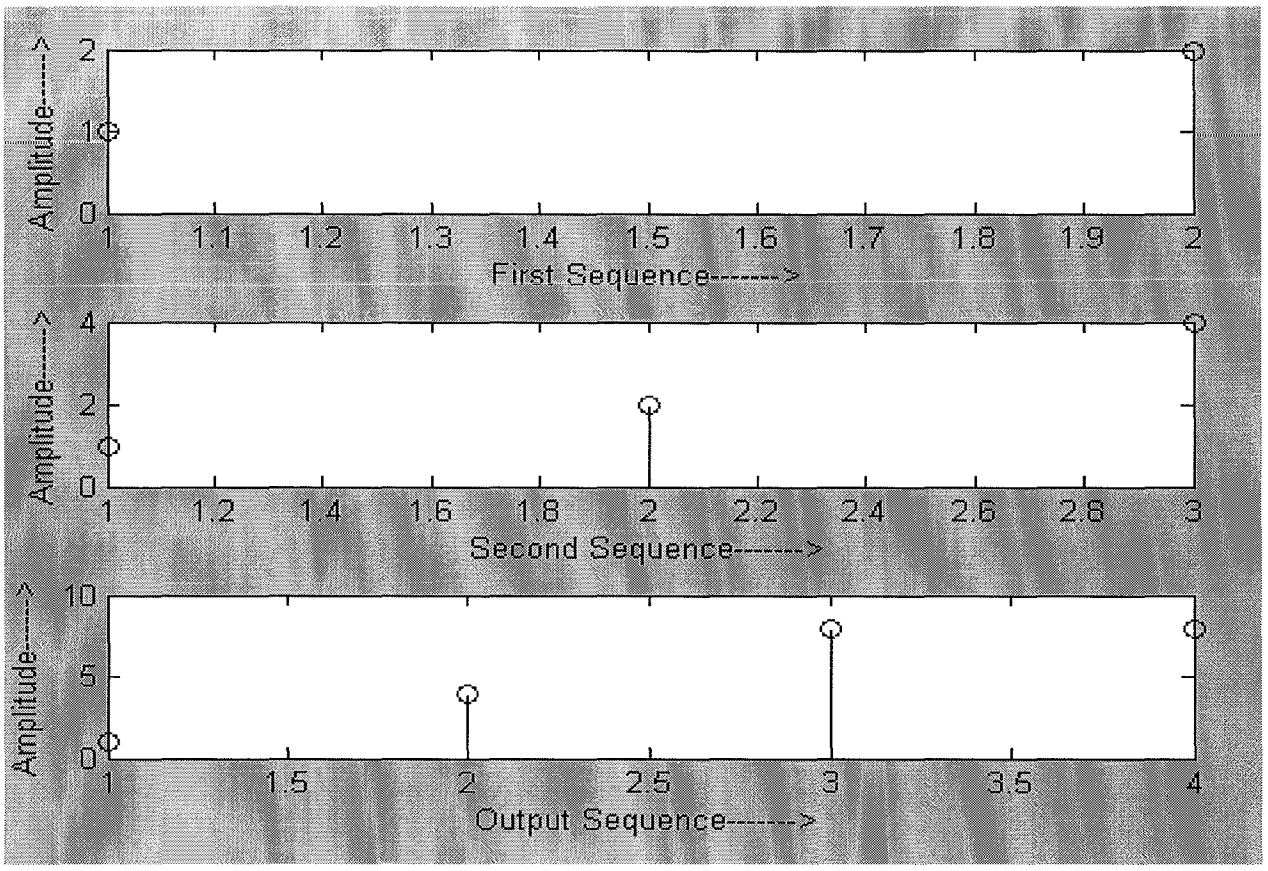
a)LINEAR CONVOLUTION

```
clc; clear all; close all;
x =input ('Enter the First Sequence: ' );
h = input ('Enter the Second Sequence: ');
y = conv(x,h);
subplot(3,1,1);stem(x);
ylabel( 'AMPLITUDE----->');
xlabel('First Sequence---->');
subplot(3,1,2);stem(h);
ylabel( 'AMPLITUDE----->');xlabel('Second Sequence----->') ;
subplot(3,1,3);stem(y);
ylabel( 'AMPLITUDE----->');xlabel('Output Sequence---->');
disp( 'The Resultant Signal is: ');y
```

(b) CIRCULAR CONVOLUTION

```
clc;clear all;close all;

% program to perform Circular Convolution

x1 = input ('Enter the First Sequence to be convoluted:');
l1 = length(x1);

x2 = input ('Enter the Second Sequence to be convoluted:');
l2 = length(x2);
l3 = l1 + l2 -1;

x1=[x1,zeros(1,l3-l1)];
x2= [x2,zeros(1,l3-l2)];

disp('The Input Squence:');x1,x2

f = cconv(x1,x2,l3);

disp('The Circular Convoluted Sequence is:');
disp(f);

subplot(311);
stem(x1);
xlabel('Time');ylabel('Amplitude');title('First Sequence');

subplot(312);
stem(x2);
xlabel('Time');ylabel('Amplitude');title('First Sequence');

subplot(313);
stem(f);
xlabel('Time');
ylabel('Amplitude');
title('Circular Convoluted sequence');
```

(a)LINEAR CONVOLUTION



Enter the First Sequence : [1 2]

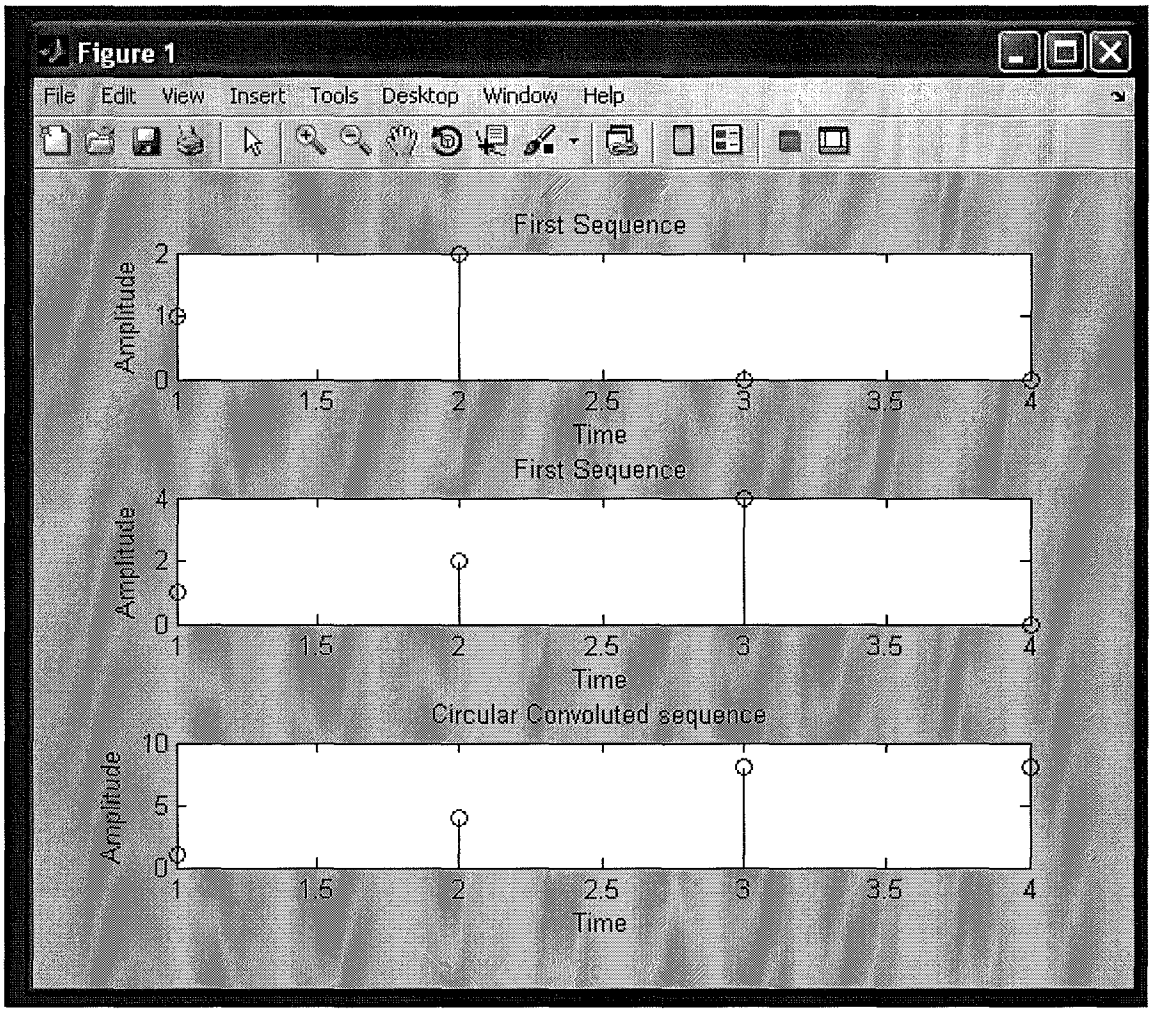
Enter the second Sequence : [1 2 4]

The Resultant signal is

Y=

1 4 8 8

(b)CIRCULAR CONVOLUTION



x1 = 1 2 0 0
x2 = 1 2 4 0

The Circular Convoluted Sequence is:
1 4 8 8

RESULT:

The Linear Convolution and Circular Convolution are implemented using MATLAB.

SPECTRUM ANALYSIS USING DFT

EXPERIMENT NO: 4 DATE:

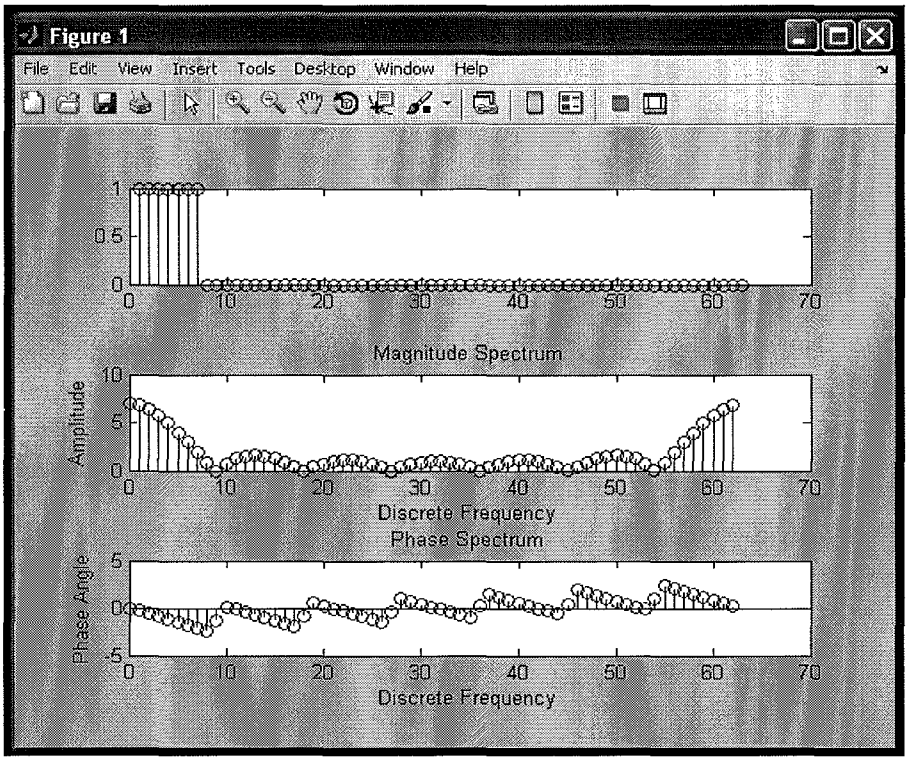
AIM:

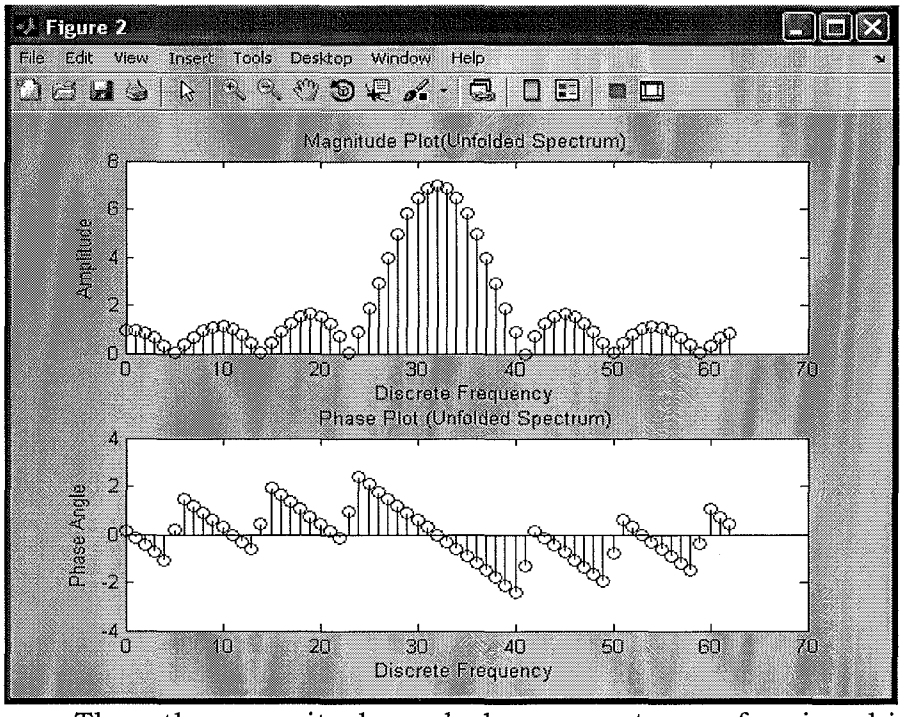
To plot the magnitude and phase spectrum of a signal using DFT.

PROGRAM:

```
% Magnitude and Phase Spectrum of a rect signal

 clc;clear all;close all;
N = input ('Length of the Sequences,  N=');
m = floor(N/8);
rect = [ones(1,m) zeros(1,N-m)];
subplot(311);stem(rect);

% Finding DFT

w = exp(-j*2*pi/N);
n = [0:1:(N-1)]; k = [0:1:(N-1)];
nk = n'*k;
W = w.^nk;
X = rect*W;

% Magnitude and Phase Plots

subplot(312);stem(k,abs(X(1:N)));
title('Magnitude Spectrum');xlabel('Discrete Frequency');
ylabel('Amplitude');subplot(313);stem(k,angle(X(1:N)));title('Phase Spectrum');
xlabel('Discrete Frequency');ylabel('Phase Angle');
disp('Press any Key to Continue');pause;

% Unfolding the Spectrum

y = X(floor(N/2)+1:N);
Y = [y X(1:floor(N/2))];
figure,subplot(211),stem(k,abs(Y(1:N)));
title('Magnitude Plot(Unfolded Spectrum)');
xlabel('Discrete Frequency');ylabel('Amplitude');
subplot(212);stem(k,angle(Y(1:N)));
title('Phase Plot (Unfolded Spectrum)');
xlabel('Discrete Frequency');ylabel('Phase Angle');
```

Length of the Sequences, N=63
Press any Key to Continue





RESULT: Thus the magnitude and phase spectrum of a signal is determined using DFT.

STUDY OF ARCHITECTURE OF DIGITAL SIGNAL PROCESSOR – TMS 320C5416

EXPERIMENT NO: 5 DATE:

AIM:

To study the architecture of digital signal processor TMS 320C5416

ARCHITECTURE:

The TMS320VC5416 fixed-point, digital signal processor (DSP) (hereafter referred to as the device unless otherwise specified) is based on an advanced modified Harvard architecture that has one program memory bus and three data memory buses. This processor provides an arithmetic logic unit (ALU) with a high degree of parallelism, application-specific hardware logic, on-chip memory, and additional on-chip peripherals. The basis of the operational flexibility and speed of this DSP is a highly specialized instruction set.

Separate program and data spaces allow simultaneous access to program instructions and data, providing a high degree of parallelism. Two read operations and one write operation can be performed in a single cycle. Instructions with parallel store and application-specific instructions can fully utilize this architecture.

In addition, data can be transferred between data and program spaces. Such parallelism supports a powerful set of arithmetic, logic, and bit-manipulation operations that can all be performed in a single machine cycle. The device also includes the control mechanisms to manage interrupts, repeated operations, and function calls.

Memory

The device provides both on-chip ROM and RAM memories to aid in system performance and integration.

Data Memory

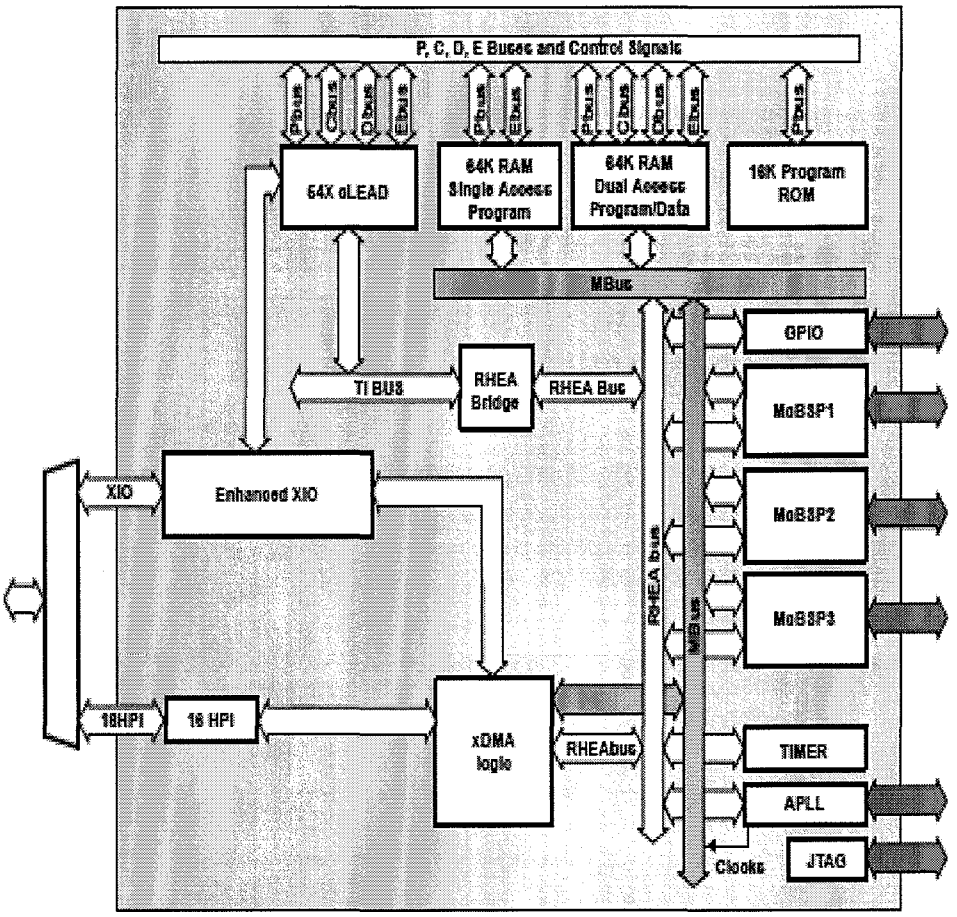
The data memory space addresses up to 64K of 16-bit words. The device automatically accesses the on-chip RAM when addressing within its bounds. When an address is generated outside the RAM bounds, the device automatically generates an external access.

The advantages of operating from on-chip memory are as follows:

· Higher performance because no wait states are required
· Higher performance because of better flow within the pipeline of the central arithmetic logic unit (CALU)
· Lower cost than external memory
· Lower power than external memory

The advantage of operating from off-chip memory is the ability to access a larger address space.

The functional overview of the processor is based on the following block diagram.



Program Memory

Software can configure their memory cells to reside inside or outside of the program address map. When the cells are mapped into program space, the device automatically accesses them when their addresses are within bounds. When the program-address generation (PAGEN) logic generates an address outside its bounds, the device automatically generates an external access. The advantages of operating from on-chip memory are as follows:

· Higher performance because no wait states are required

· Lower cost than external memory
· Lower power than external memory

The advantage of operating from off-chip memory is the ability to access a larger address space.

Extended Program Memory

The device uses a paged extended memory scheme in program space to allow access of up to 8192K of program memory. In order to implement this scheme, the device includes several features which are also present on C548/549/5410:

· Twenty-three address lines, instead of sixteen
· An extra memory-mapped register, the XPC
· Six extra instructions for addressing extended program space

Program memory in the device is organized into 128 pages that are each 64K in length. The value of the XPC register defines the page selection. This register is memory-mapped into data space to address 001Eh. At a hardware reset, the XPC is initialized to 0.

On-Chip Peripherals

The device has the following peripherals:

· Software-programmable wait-state generator
· Programmable bank-switching
· A host-port interface (HPI8/16)
· Three multichannel buffered serial ports (McBSPs)
· A hardware timer
· A clock generator with a multiple phase-locked loop (PLL)
· Enhanced external parallel interface (XIO2)
· A DMA controller (DMA)

Parallel I/O Ports

The device has a total of 64K I/O ports. These ports can be addressed by the PORTR instruction or the PORTW instruction. The IS signal indicates a read/write operation through an I/O port. The device can interface easily with external devices through the I/O ports while requiring minimal off-chip address-decoding circuits.

Multichannel Buffered Serial Ports (McBSPs)

The device provides three high-speed, full-duplex, multichannel buffered serial ports that allow direct interface to other C54x/LC54x devices, codecs, and other devices in a system. The McBSPs are based on the standard serial-

port interface found on other 54x devices. Like their predecessors, the McBSPs provide:
· Full-duplex communication
· Double-buffer data registers, which allow a continuous data stream
· Independent framing and clocking for receive and transmit
In addition, the McBSPs have the following capabilities:
· Direct interface to:
– T1/E1 framers
– MVIP switching compatible and ST-BUS compliant devices
– IOM-2 compliant devices
– AC97-compliant devices
– IIS-compliant devices
– Serial peripheral interface
· Multichannel transmit and receive of up to 128 channels
· A wide selection of data sizes, including 8, 12, 16, 20, 24, or 32 bits
· m-law and A-law companding
· Programmable polarity for both frame synchronization and data clocks
· Programmable internal clock and frame generation.

Hardware Timer

The device features a 16-bit timing circuit with a 4-bit prescaler. The timer counter is decremented by one every CLKOUT cycle. Each time the counter decrements to 0, a timer interrupt is generated. The timer can be stopped, restarted, reset, or disabled by specific status bits.

Clock Generator

The clock generator provides clocks to the device, and consists of a phase-locked loop (PLL) circuit. The clock generator requires a reference clock input, which can be provided from an external clock source. The reference clock input is then divided by two (DIV mode) to generate clocks for the device, or the PLL circuit can be used (PLL mode) to generate the device clock by multiplying the reference clock frequency by a scale factor, allowing use of a clock source with a lower frequency than that of the CPU. The PLL is an adaptive circuit that, once synchronized, locks onto and tracks an input clock signal.

DMA Controller

The device direct memory access (DMA) controller transfers data between points in the memory map without intervention by the CPU. The DMA allows movements of data to and from internal program/data memory, internal peripherals (such as the McBSPs), or external memory devices to occur in the background of CPU operation. The DMA has six independent programmable channels, allowing six different contexts for DMA operation.

The DMA has the following features:

· The DMA operates independently of the CPU.

· The DMA has six channels. The DMA can keep track of the contexts of six independent block transfers.

· The DMA has higher priority than the CPU for both internal and external accesses.

· Each channel has independently programmable priorities.

· Each channel's source and destination address registers can have configurable indexes through memory on each read and write transfer, respectively. The address may remain constant, be post-incremented, be post-decremented, or be adjusted by a programmable value.

· Each read or write internal transfer may be initialized by selected events.

· On completion of a half- or entire-block transfer, each DMA channel may send an interrupt to the CPU.

· The DMA can perform double-word internal transfers (a 32-bit transfer of two 16-bit words).

RESULT:

Thus the architecture of TMS320C5416 Processor was illustrated.

LINEAR CONVOLUTION USING TMS 320C5416 PROCESSOR

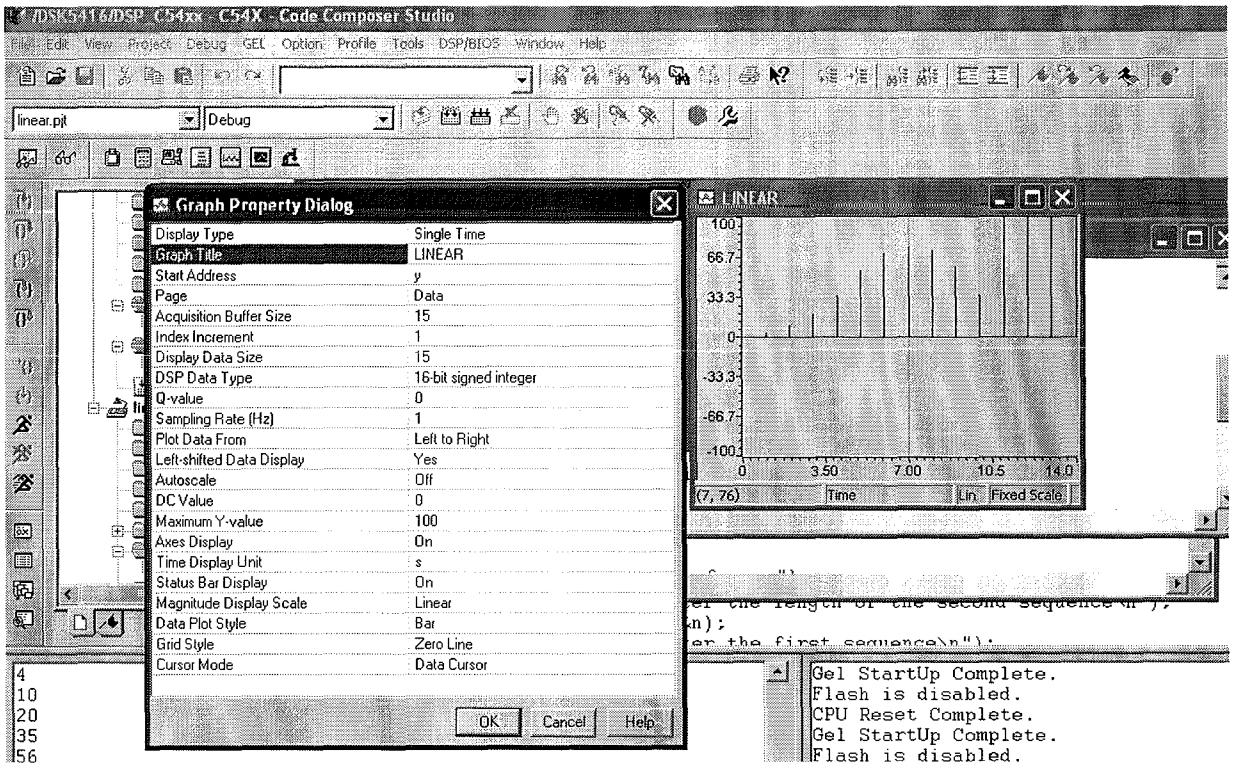
EXPERIMENT NO: 6 DATE:

AIM:

To implement Linear Convolution using TMS 320C5416 Processor.

PROGRAM:

```
#include<stdio.h>
#include<math.h>
int y[20];

main()
{
        int m =6;
        int n =6;
        int i =0;
        int j;
        int x[15] = {1,2,3,4,5,6,0,0,0,0,0,0};
        int h[15] = {1,2,3,4,5,6,0,0,0,0,0,0};
        for (i=0;i<m+n-1;i++)
        {
                y[i] =0;
                for(j=0;j<=i;j++)
                        y[i]+=x[j]*h[i-j];
        }
        for(i=0;i<m+n-1;i++)
        printf("%d\n",y[i]);

}
```



RESULT:

Thus the Linear Convolution program was implemented using TMS 320C5416 Processor.

CIRCULAR CONVOLUTION USING TMS 320C5416 PROCESSOR

EXPERIMENT NO: 7 DATE:

AIM:

To implement circular Convolution using TMS 320C5416 Processor.

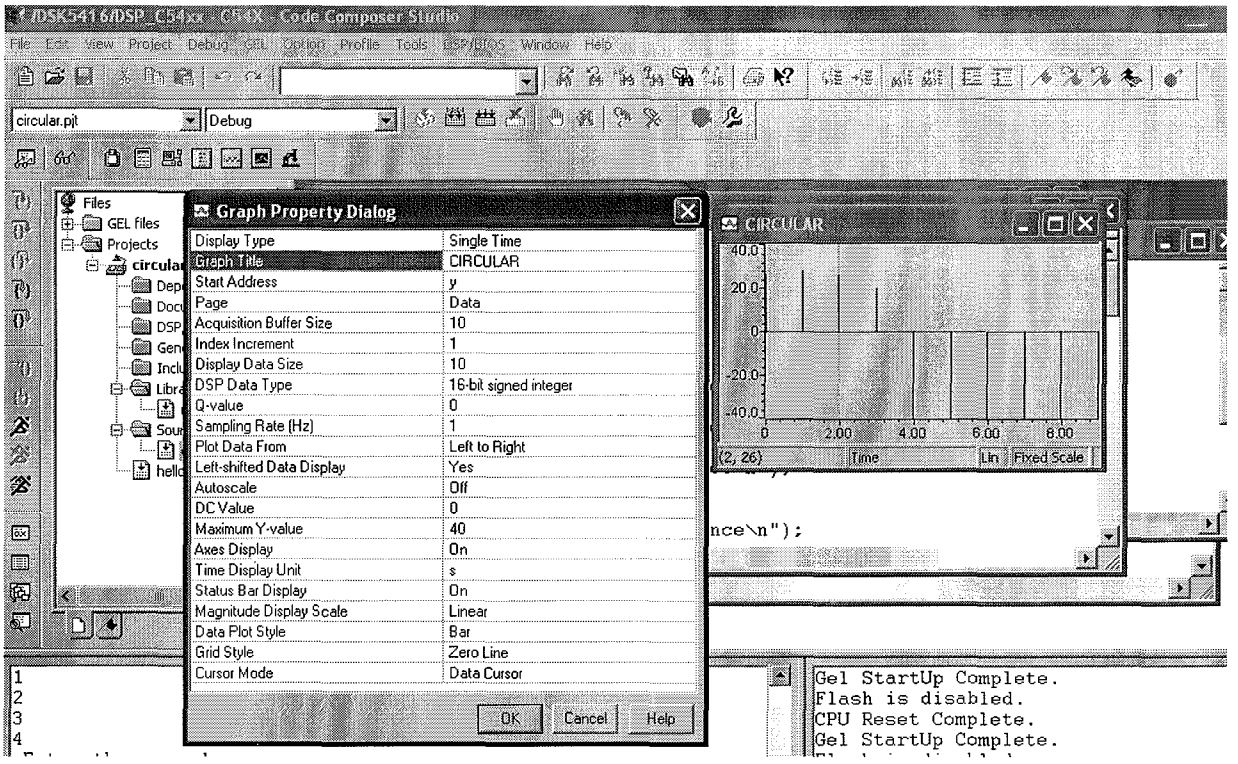
PROGRAM:

```
#include<stdio.h>
int m,n,x[30],h[30],y[30],i,j, k,x2[30],a[30];
void main()
{
        printf(" Enter the length of the first sequence\n");
        scanf("%d",&m);
        printf(" Enter the length of the second sequence\n");
        scanf("%d",&n);
        printf(" Enter the first sequence\n");
        for(i=0;i<m;i++)
                scanf("%d",&x[i]);
                printf(" Enter the second sequence\n");
                for(j=0;j<n;j++)
                        scanf("%d",&h[j]);
                        if(m-n!=0) /*If length of both sequences are not equal*/
                        {
                                if(m>n) /* Pad the smaller sequence with zero*/
                                {
                                        for(i=n;i<m;i++)
                                        h[i]=0;
                                        n=m;
                                }
                                for(i=m;i<n;i++)
                                x[i]=0;
                                m=n;
                        }
                        y[0]=0;
                        a[0]=h[0];
                        for(j=1;j<n;j++) /*folding h(n) to h(-n)*/
                        a[j]=h[n-j];
                        /*Circular convolution*/
                        for(i=0;i<n;i++)
                                y[0]+=x[i]*a[i];
                                for(k=1;k<n;k++)
```

```
                {
                        y[k]=0;
                /*circular shift*/
                        for(j=1;j<n;j++)
                        x2[j]=a[j-1];
                        x2[0]=a[n-1];
                        for(i=0;i<n;i++)
                        {
                                a[i]=x2[i];
                                y[k]+=x[i]*x2[i];
                        }
                }
        /*displaying the result*/
    printf(" The circular convolution is\n");
    for(i=0;i<n;i++)
    printf("%d \t",y[i]);
}
```

As an Example:

```
Enter the length of the first sequence
4
Enter the length of the second sequence
4
Enter the first sequence
1 2 3 4
Enter the second sequence
1 2 3 4
The circular convolution is
26 28 26 20
```



RESULT:

Thus the Circular Convolution was implemented using TMS 320C5416 Processor.

WAVEFORM GENERATION USING TMS 320C5416

EXPERIMENT NO: 8 DATE:

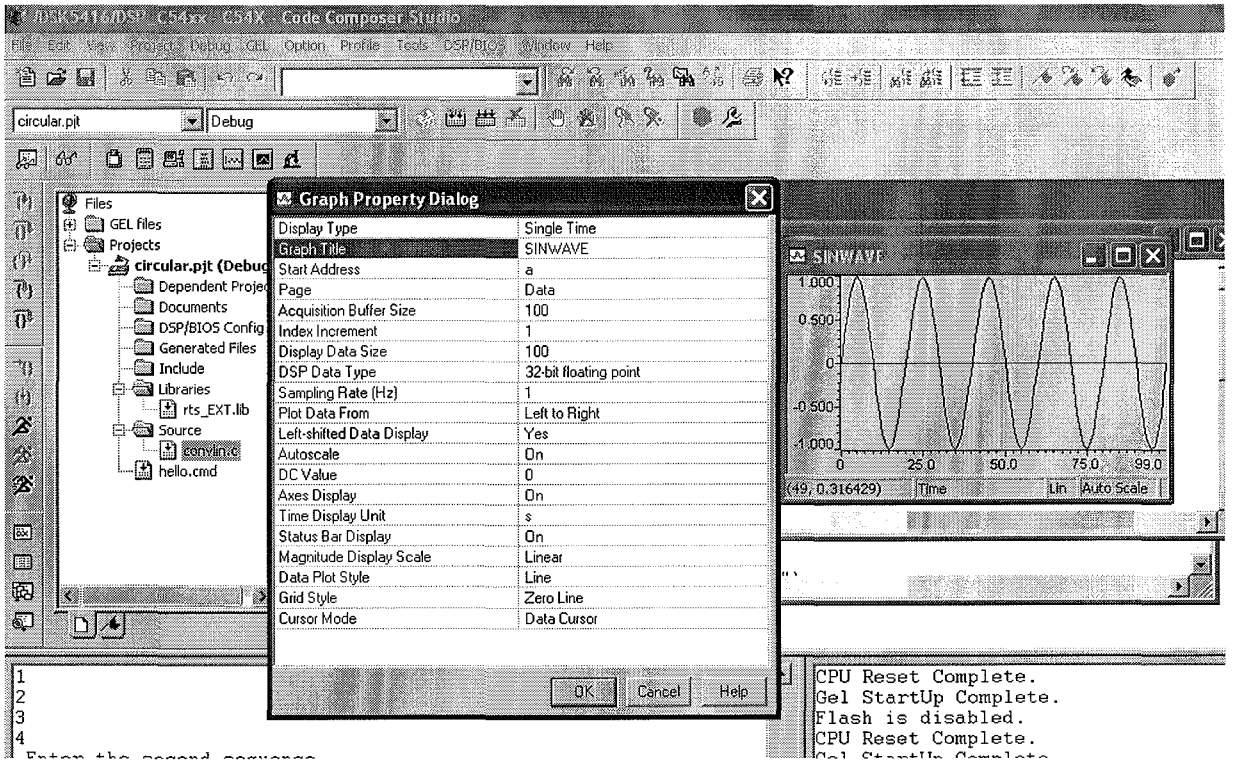
AIM:

To generate sine and cosine waveform using TMS320C5416 Processor.

PROGRAM:

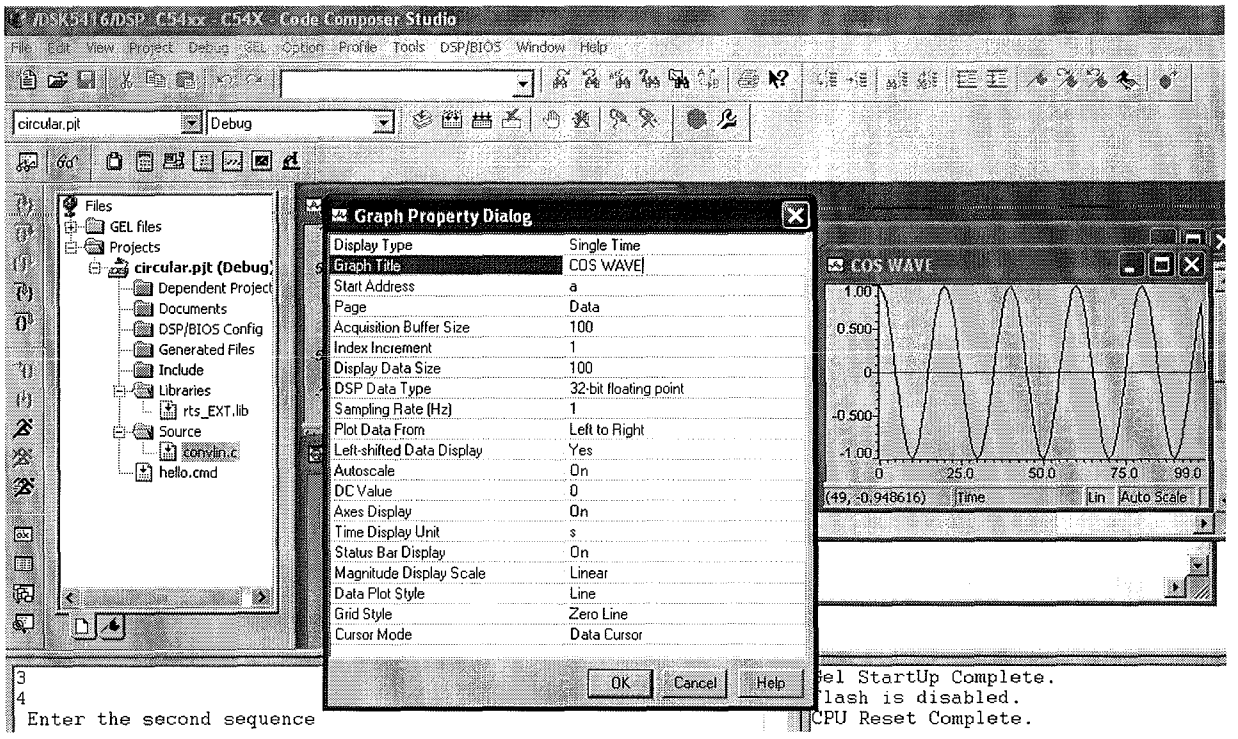
SIN.C

```
# include <stdio.h>
# include <math.h>
float a[100];
main()
{
        int i;
        for(i=0;i<99;i++)
        {
                a[i] = sin(2*3.14*5*i/100);
                printf("%f",a[i]);
        }
}
```



COS.C

```
# include <stdio.h>
# include <math.h>
float a[100];
main()
{
        int i;
        for(i=0;i<99;i++)
        {
                a[i] = cos(2*3.14*5*i/100);
                printf("%f",a[i]);
        }
}
```



RESULT:

Thus the waveforms were generated using TMS320C5416 Processor.

CYCLE – II

DESIGN OF FIR FILTER

EXPERIMENT NO: 1 DATE:

AIM:

To design FIR Low pass filter, FIR High-pass Filter, FIR Band-pass Filter using Rectangular , Black man, Hamming and Hanning window Techniques using MATLAB.

FORMULA:

$$N = \frac{-20\,log(\sqrt{rp * rs} - 13}{14.6(fs - fp)/f}$$

PROGRAM:

(A) RECTANGULAR WINDOW (BOXCAR)

% PROGRAM FOR THE DESIGN OF FIR LPF, HPF, BPF & BSF USING RECTANGULAR WINDOW

```
clc; close all; clear all;
rp = input('Enter the Pass Band Ripple : ');
rs = input('Enter the Stop Band Ripple : ');
fp = input('Enter the Pass Band Frequency: ');
fs = input('Enter the Stop Band Frequency: ');
f = input('Enter the Sampling Frequency : ');
wp = 2 * fp/f;
ws = 2 * fs/f;
num = - 20 * log10( sqrt(rp*rs))- 13;
den = 14.6 * (fs-fp)/f;
n = ceil (num/den) ;
n1 = n+1;
if (rem(n,2)~=0) ;
n1 = n;
n = n-1;
end
y = boxcar (n1) ;
```

```
% LOW PASS FILTER

b = fir1(n,wp,y);
[h,o] = freqz(b,1,256);
m = 20 * log10(abs(h));
subplot(2,2,1) ;plot (o/pi,m) ;
title(' ***** RECTANGULAR WINDOW or BOXCAR *****');
ylabel('Gain indb------>');xlabel(' (a) Normalised Frequency------->');


% HIGH PASS FILTER
b = fir1(n,wp,'high',y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,2) ;plot(o/pi,m) ;
ylabel('Gain in db------>');xlabel(' (b) Normalised Frequency------->');

% BAND PASS FILTER
wn = [wp ws]; b = fir1(n,wn,y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,3) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (c) Normalised Frequency------->');

% BAND STOP FILTER
b = fir1 (n, wn, 'stop' , y) ;
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,4) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (d) Normalised Frequency------->');
```

RECTANGULAR WINDOW

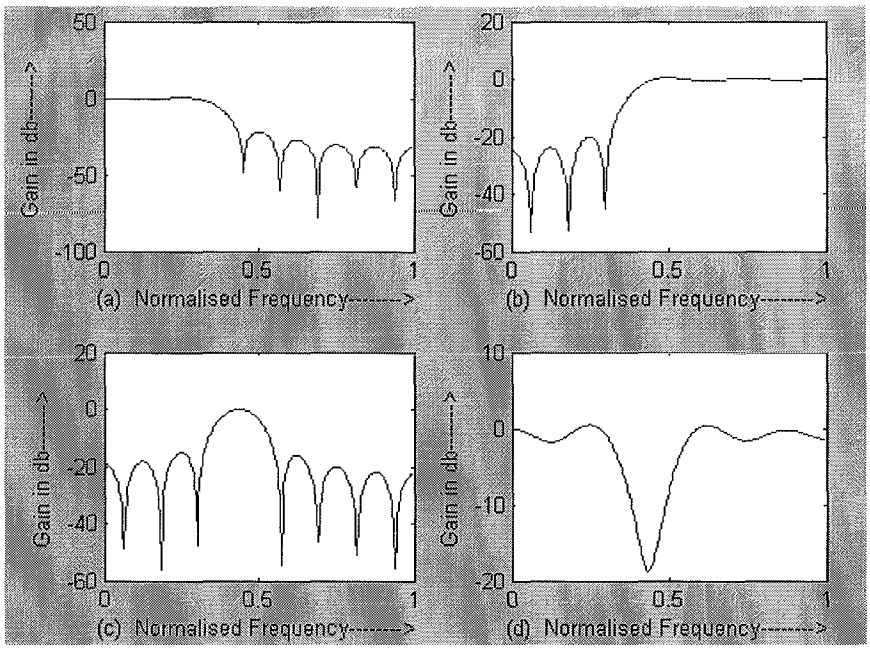
Enter the Pass Band Ripple : 0.05

Enter the Stop Band Ripple : 0.04

Enter the Pass Band Frequency: 1500

Enter the Stop Band Frequency: 2000

Enter the Sampling Frequency : 8000



(B) BLACK MAN WINDOW

% PROGRAM FOR THE DESIGN OF FIR LPF, HPF, BPF & BSF USING BLACKMAN WINDOW

```
clc; close all; clear all;
rp = input('Enter the Pass Band Ripple : ');
rs = input('Enter the Stop Band Ripple : ');
fp = input('Enter the Pass Band Frequency: ');
fs = input('Enter the Stop Band Frequency: ');
f = input('Enter the Sampling Frequency : ');
wp = 2 * fp/f;
ws = 2 * fs/f;
num = - 20 * log10( sqrt(rp*rs))- 13;
den = 14.6 * (fs-fp)/f;
n = ceil (num/den) ;
n1 = n+1;
if (rem(n,2)~=0) ;
n1 = n;
n = n-1;
end
y = blackman (n1) ;
```

```
% LOW PASS FILTER

b = fir1(n,wp,y);
[h,o] = freqz(b,1,256);
m = 20 * log10(abs(h));
subplot(2,2,1) ;plot (o/pi,m) ;
title(' ***** BLACKMAN WINDOW *****');
ylabel('Gain indb------>');xlabel(' (a) Normalised Frequency------->');

% HIGH PASS FILTER

b = fir1(n,wp,'high',y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,2) ;plot(o/pi,m) ;
ylabel('Gain in db------>');xlabel(' (b) Normalised Frequency------->');

% BAND PASS FILTER

wn = [wp ws]; b = fir1(n,wn,y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,3) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (c) Normalised Frequency------->');

% BAND STOP FILTER

b = fir1 (n, wn, 'stop' , y) ;
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,4) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (d) Normalised Frequency------->');
```

BLACKMAN WINDOW

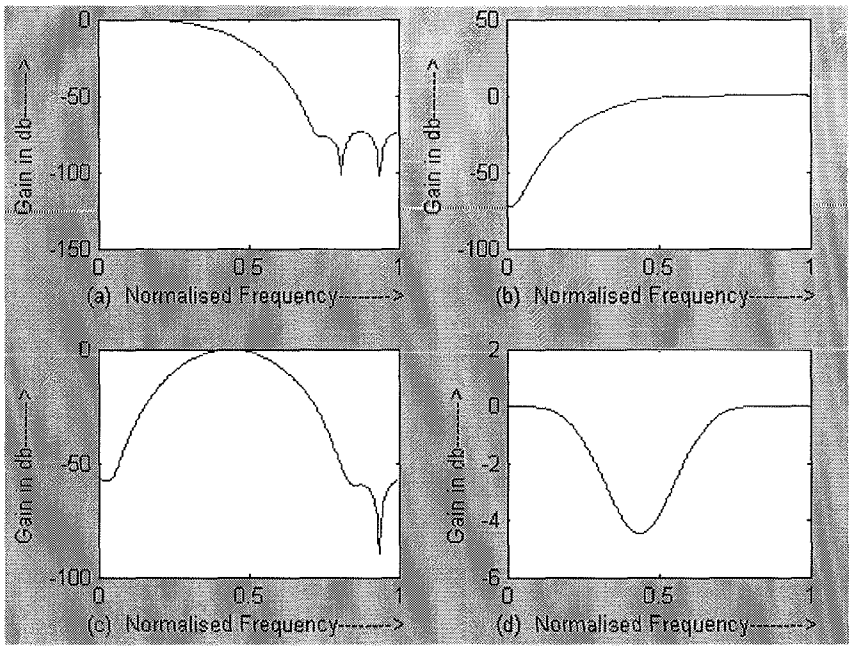
Enter the Pass Band Ripple : 0.05

Enter the Stop Band Ripple : 0.04

Enter the Pass Band Frequency: 1500

Enter the Stop Band Frequency: 2000

Enter the Sampling Frequency : 8000



(C) HAMMING WINDOW

```
% PROGRAM FOR THE DESIGN OF FIR LPF, HPF, BPF & BSF USING
HAMMING WINDOW
clc; close all; clear all;
rp = input('Enter the Pass Band Ripple : ');
rs = input('Enter the Stop Band Ripple : ');
fp = input('Enter the Pass Band Frequency: ');
fs = input('Enter the Stop Band Frequency: ');
f = input('Enter the Sampling Frequency : ');
wp = 2 * fp/f;
ws = 2 * fs/f;
num = - 20 * log10( sqrt(rp*rs))- 13;
den = 14.6 * (fs-fp)/f;
n = ceil (num/den) ;
n1 = n+1;
if (rem(n,2)~=0) ;
n1 = n;
n = n-1;
end
y = hamming (n1) ;
```

```
% LOW PASS FILTER

b = fir1(n,wp,y);
[h,o] = freqz(b,1,256);
m = 20 * log10(abs(h));
subplot(2,2,1) ;plot (o/pi,m) ;
title(' ***** HAMMING WINDOW *****');
ylabel('Gain indb------>');xlabel(' (a) Normalised Frequency------->');

% HIGH PASS FILTER

b = fir1(n,wp,'high',y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,2) ;plot(o/pi,m) ;
ylabel('Gain in db------>');xlabel(' (b) Normalised Frequency------->');

% BAND PASS FILTER

wn = [wp ws]; b = fir1(n,wn,y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,3) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (c) Normalised Frequency------->');

% BAND STOP FILTER

b = fir1 (n, wn, 'stop' , y) ;
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,4) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (d) Normalised Frequency------->');
```

HAMMING WIDOW

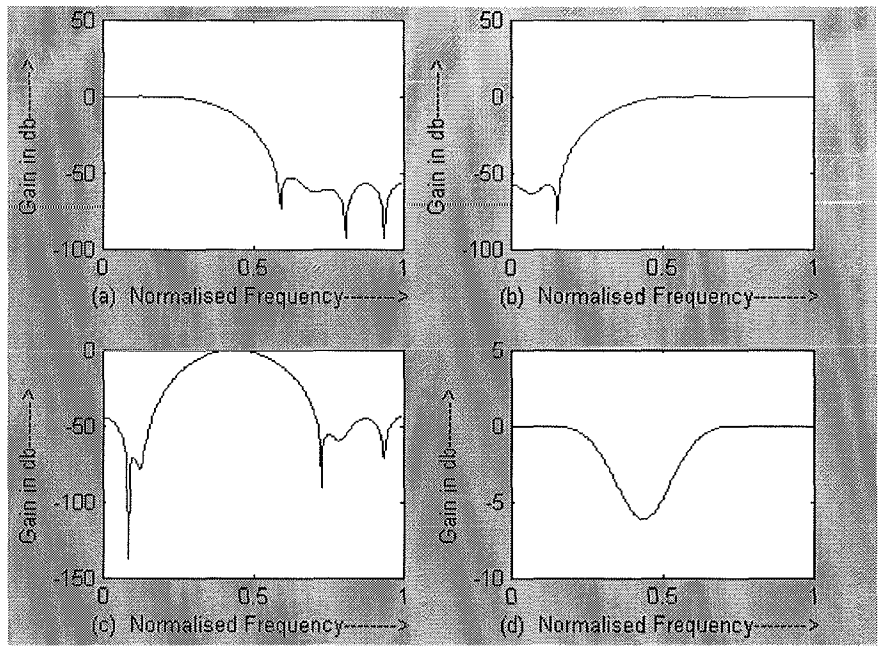
Enter the Pass Band Ripple : 0.05

Enter the Stop Band Ripple : 0.04

Enter the Pass Band Frequency: 1500

Enter the Stop Band Frequency: 2000

Enter the Sampling Frequency : 8000



(D) HANNING WINDOW

```
% PROGRAM FOR THE DESIGN OF FIR LPF, HPF, BPF & BSF USING
HANNING WINDOW %
clc; close all; clear all;
rp = input('Enter the Pass Band Ripple : ');
rs = input('Enter the Stop Band Ripple : ');
fp = input('Enter the Pass Band Frequency: ');
fs = input('Enter the Stop Band Frequency: ');
f = input('Enter the Sampling Frequency : ');
wp = 2 * fp/f;
ws = 2 * fs/f;
num = - 20 * log10( sqrt(rp*rs))- 13;
den = 14.6 * (fs-fp)/f;
n = ceil (num/den) ;
n1 = n+1;
if (rem(n,2)~=0) ;
n1 = n;
n = n-1;
end
y = hanning (n1) ;
```

```
% LOW PASS FILTER

b = fir1(n,wp,y);
[h,o] = freqz(b,1,256);
m = 20 * log10(abs(h));
subplot(2,2,1) ;plot (o/pi,m) ;
title(' ***** HANNING WINDOW *****');
ylabel('Gain indb------>');xlabel(' (a) Normalised Frequency------->');

% HIGH PASS FILTER

b = fir1(n,wp,'high',y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,2) ;plot(o/pi,m) ;
ylabel('Gain in db------>');xlabel(' (b) Normalised Frequency------->');

% BAND PASS FILTER

wn = [wp ws]; b = fir1(n,wn,y);
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,3) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (c) Normalised Frequency------->');

% BAND STOP FILTER

b = fir1 (n, wn, 'stop' , y) ;
[h,o] = freqz(b,1,256);
m = 20*log10(abs(h));
subplot(2,2,4) ;plot (o/pi, m) ;
ylabel('Gain in db------>');xlabel(' (d) Normalised Frequency------->');
```

HANNING WINDOW

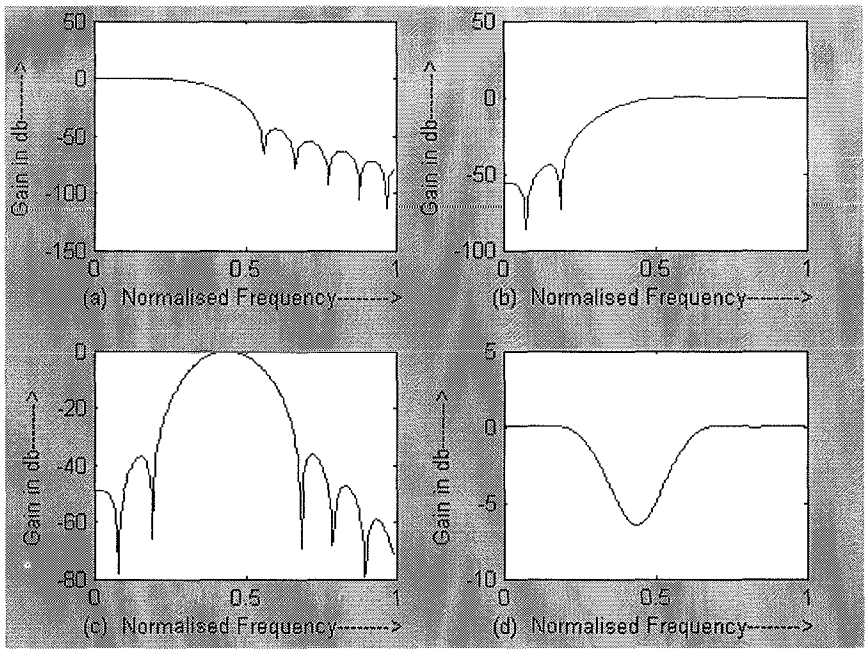
Enter the Pass Band Ripple : 0.05

Enter the Stop Band Ripple : 0.04

Enter the Pass Band Frequency: 1500

Enter the Stop Band Frequency: 2000

Enter the Sampling Frequency : 8000



RESULT:

Thus FIR - LPF, HPF, BPF, BSF using Rectangular, Blackman, Hamming and Hanning Window Technique was implemented using MATLAB.

IIR FILTER DESIGN

EXPERIMENT NO: 2 DATE:

AIM:

To design butterworth IIR Filter using MATLAB Program.

PROGRAM: (a)

```
clc;clear all;close all;
rp=input('Enter the Passband ripple: ');
rs=input('Enter the Stopband filter: ');
wp=input('Enter Passband frequency: ');
ws=input('Enter Stopband frequency: ');
fs=input('Enter Sampling frequency: ');
w1=2*wp/fs;w2=2*ws/fs;
[n,wn]=buttord(w1,w2,rp,rs) ;
[b,a]=butter(n,wn) ;
w=0:.01:pi; [h,om]=freqz(b,a,w);
m=20*log10(abs(h)) ;
an=angle(h) ;subplot(2,1,1) ;plot(om/pi,m);
title('IIR FILTER USING IMPULSE INVARIENT METHOD');
ylabel('Gain in db-->') ;xlabel(' (a) Normalised Frequency-->');
subplot (2,1,2) ;plot(om/pi,an);
xlabel(' (b) Normalised Frequency-->') ;ylabel('Phase in Radian-->');
```

As an Example Value

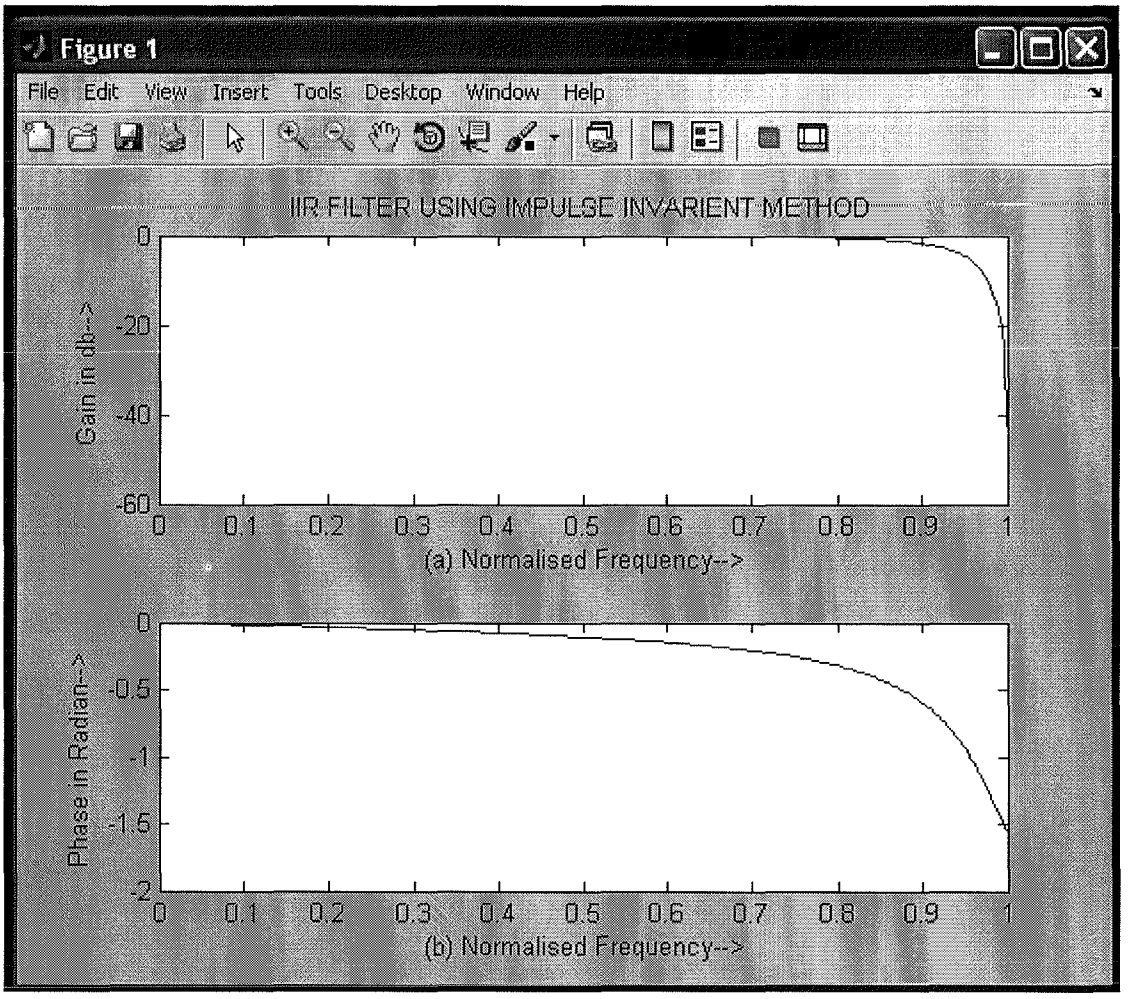
Enter the Passband ripple: 0.04

Enter the Stopband filter: 0.05

Enter Passband frequency: 1500

Enter Stopband frequency: 2000

Enter Sampling frequency: 8000



PROGRAM: (b)

```
clc;close all;clear all;
b  = input('Enter the B value     :');
a  = input('Enter the A value     :');
fs = input('Enter the Frequency   :');
[bz az] = bilinear(b,a,fs);
[bz1 az1] = impinvar(b,a,fs);
disp('The Output is :');bz ,az,bz1 ,az1
```

Enter the B value :4

Enter the A value :[1 2.828 4]

Enter the Frequency :1

The Output is :

bz = 0.2929 0.5858 0.2929

az = 1.0000 0.0000 0.1716

bz1 = 0 0.6793

az1 = 1.0000 -0.0757 0.0591

RESULT:

Thus the IIR Filter was designed using Matlab.

MULTIRATE FILTERS

EXPERIMENT NO: 3 DATE:

AIM:

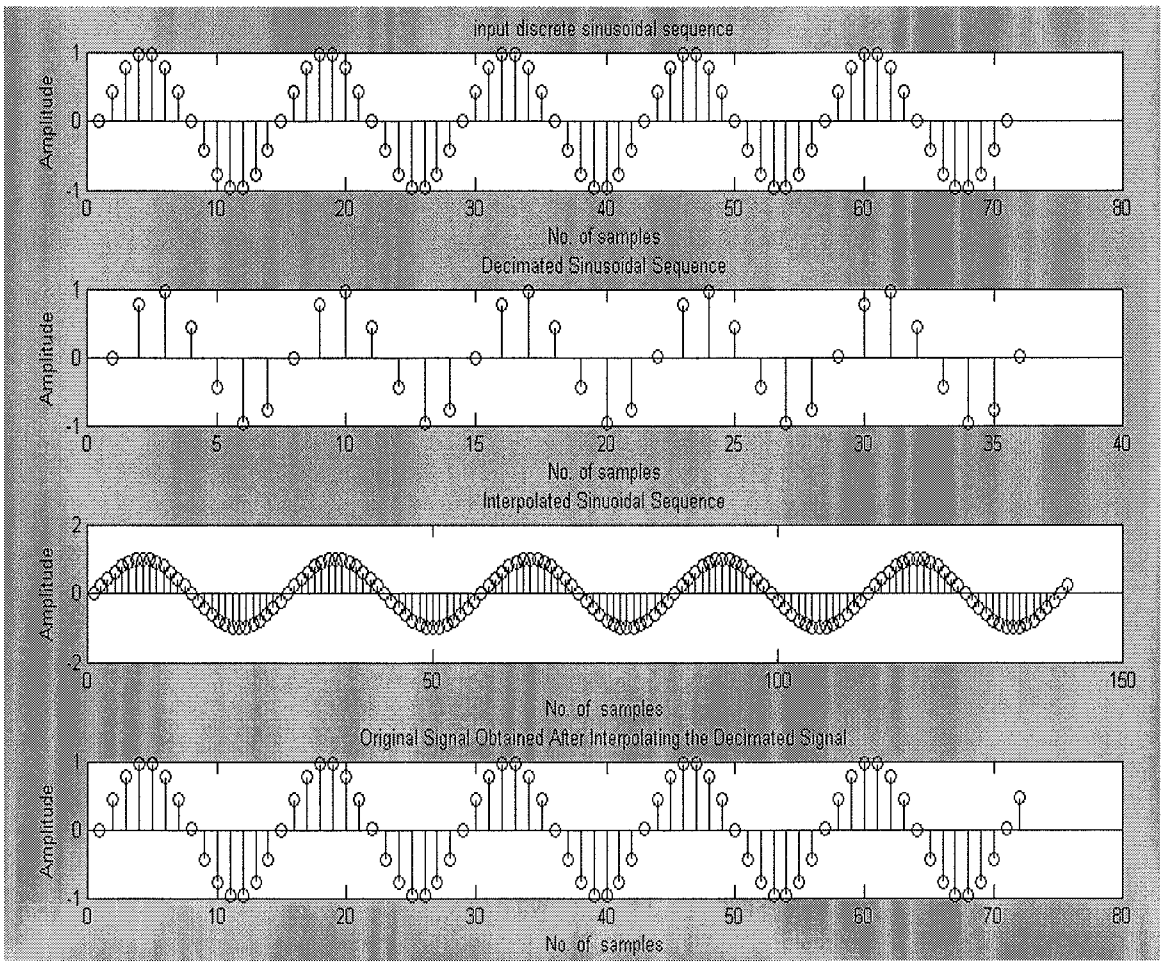
To perform interpolation and decimation using MATLAB.

(A) INTERPOLATION AND DECIMATION

PROGRAM:

```
clc;                        % clears the command window
close all;                  % closes all the previously open window
clear all;                  % clears previously stored values

                % Generating Input Sequence
fm = 10;                    %  input signal frequency
Fs = 140;                   %  sampling frequency
t = 0:1/Fs:0.5;             % time range for the input sequence
x = sin(2*pi*fm*t);         % input sinusoidal signal
figure(1)
subplot(4,1,1)
stem(x);                    % Discrete plot of the input sequence,...
                    ... where x-axis corresponds to the number of samples
xlabel('No. of samples');  % labelling x-axis
ylabel('Amplitude');        % labelling y-axis
title('input discrete sinusoidal sequence'); % giving title to the plot

                            % Decimation of the Input Sequence
M = 2;                      % factor by which the input sequence is decimated
xd = decimate(x,M);         % resamples the sample in x with a rate (1/M) times
the original rate
subplot(4,1,2)
stem(xd)                    % Discrete plot of the input sequence,...
                ... where x-axis corresponds to the number of samples
xlabel('No. of samples');   % labelling x-axis
ylabel('Amplitude');        % labelling y-axis
title('Decimated Sinusoidal Sequence'); % giving title to the plot


                        % Interolation of the Input Sequence
L = 2;                  % factor by which the input sequence is interpolated
xI = interp(x,L);       % resamples the sample in x with a rate L times the original
rate
subplot(4,1,3);
```

```
stem(xI);              % Discrete plot of the input sequence,...
                  ... where x-axis corresponds to the number of samples
xlabel('No. of  samples');          % laeblling x-axis
ylabel('Amplitude');                % labelling y-axis
title('Interpolated Sinuoidal Sequence')      % giving title to the plot

                % Interpolation of the Decimated Signal
L = 2;                % coefficient by which the singal is interpolated
xI = interp(xd,L);  % resamples the sample in x with a rate L times the original
rate
subplot(4,1,4)
stem(xI);              % Discrete plot of the input sequence,...
                  ... where x-axis corresponds to the number of samples
xlabel('No. of  samples'); % labelling x-axis
ylabel('Amplitude');       % labelling y-axis
title('Original Signal Obtained After Interpolating the Decimated Signal'); %
giving title to the graph
```

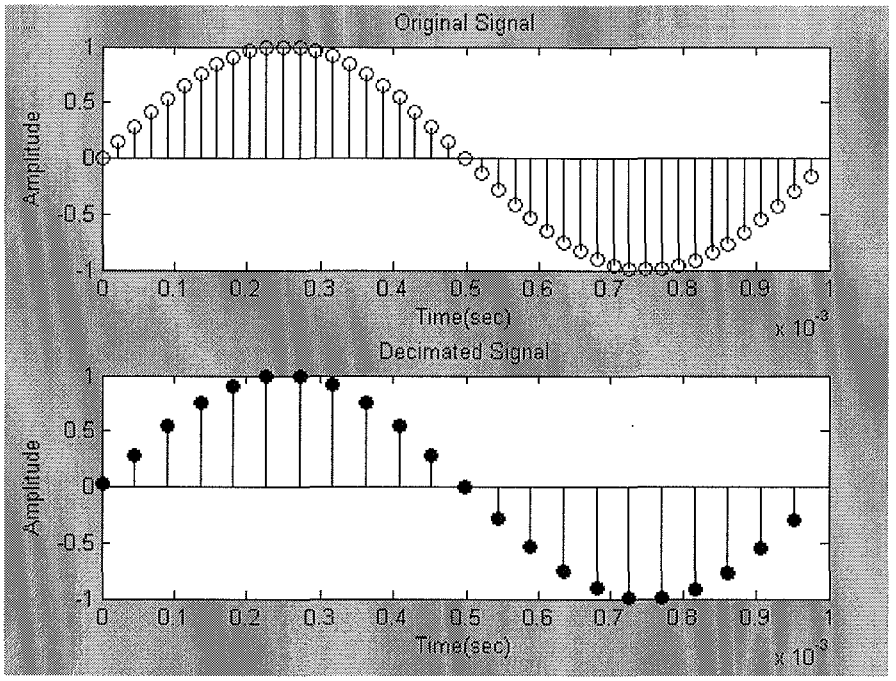


(B) DECIMATION BY POLYPHASE DECOMPOSITION

PROGRAM:

```
clc;clear all;close all;
M = 2;                          % Decimation factor
Hm =mfilt.firdecim(M);          % We use the default filter
Fs = 44.1e3;                    % Original sampling frequency: 4401kHz
n = 0:10239;                    % 10240 samples, 00232 second long signal
x = sin(2*pi*1e3/Fs*n);         % Original signal, sinusoid at 1kHz
y = filter(Hm,x);               % 5120 samples, still 00232 seconds
subplot(2,1,1);                 % Plot original sampled at 44.1kHz
stem(n(1:44)/Fs,x(1:44)); xlabel('Time (sec) ') ;ylabel('Amplitude ');
title('Original Signal');subplot(2,1,2) ;
stem(n(1:22)/(Fs/M) ,y(13:34), 'r', 'filled');% Plot decimated signal(22.05kHz) in red
xlabel('Time (sec) ') ;ylabel('Amplitude');title('Decimated Signal');
```

DECIMATION BY POLYPHASE DECOMPOSITION



RESULT:

Thus interpolation and decimation was performed using MATLAB.

EQUALIZATION

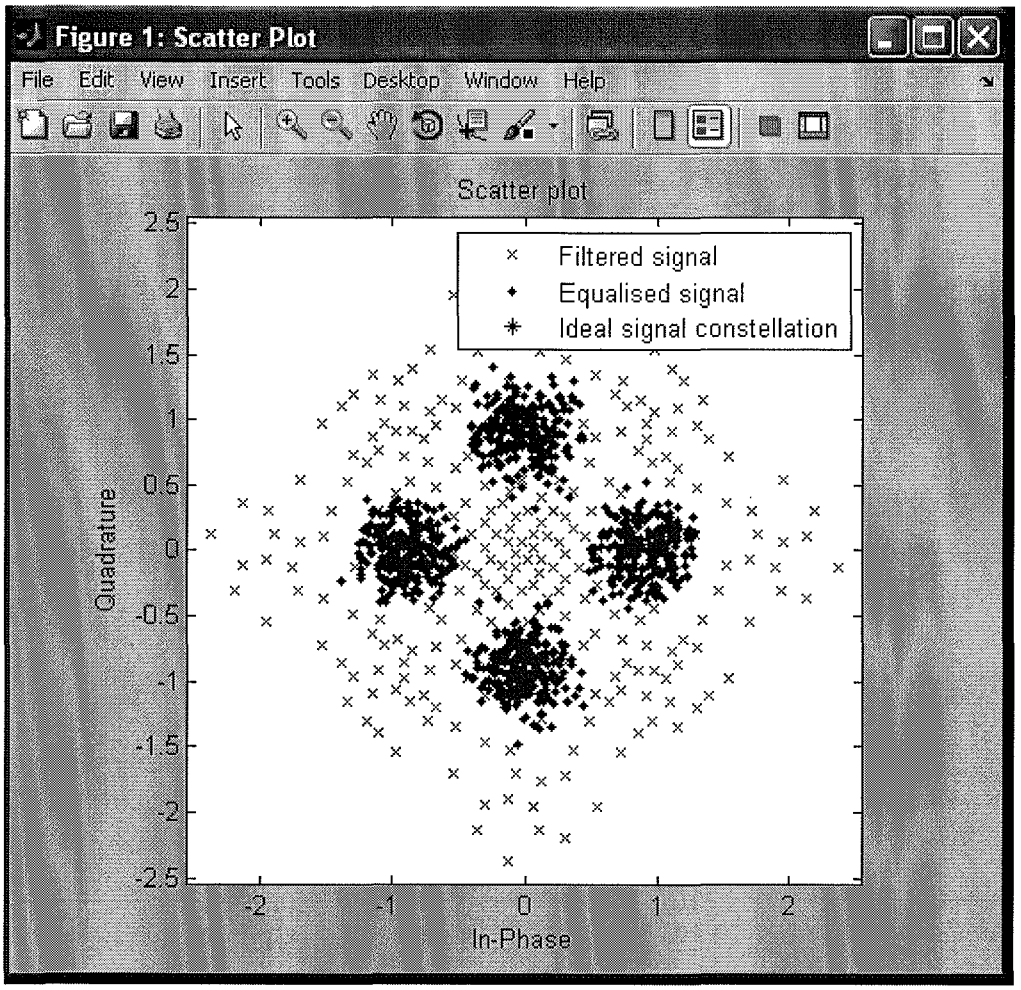
EXPERIMENT NO: 4 DATE:

AIM:

To implement Equalization using MATLAB

PROGRAM:

```
clc;clear all;close all;
M=4;
msg = randint(1500,1,M);
modmsg = pskmod(msg,M);
sigconst=pskmod([0:M-1],M);
trainlen=500;
chan= [.986;.845;.237;.123+.31i];
filtmsg=filter(chan,1,modmsg);
eqobj=lineareq(8,lms(0.01),sigconst,1);
[symbolest,yd]=equalize(eqobj,filtmsg,modmsg(1:trainlen));
h =scatterplot(filtmsg,1,trainlen,'bx');
hold on;
scatterplot(symbolest,1,trainlen,'r.',h);
scatterplot(sigconst,1,0,'k*',h);
legend('Filtered signal','Equalised signal','Ideal signal constellation');
 hold off;
demodmsg_noeq=pskdemod(filtmsg,M);
demodmsg=pskdemod(yd,M);
[nnoeq,rnoeq]=symerr(demodmsg_noeq(trainlen+1:end),msg(trainlen+1:end));
[neq,req]=symerr(demodmsg(trainlen+1:end),msg(trainlen+1:end));
disp('Symbol error rate with equaliser:');disp(req)
disp('Symbol error rate without equaliser:');disp(rnoeq)
```



RESULT:

Thus equalization was implemented using MATLAB.

IIR AND FIR IMPLEMENTATION USING TMS 320C5416

EXPERIMENT NO: 5 DATE:

AIM:

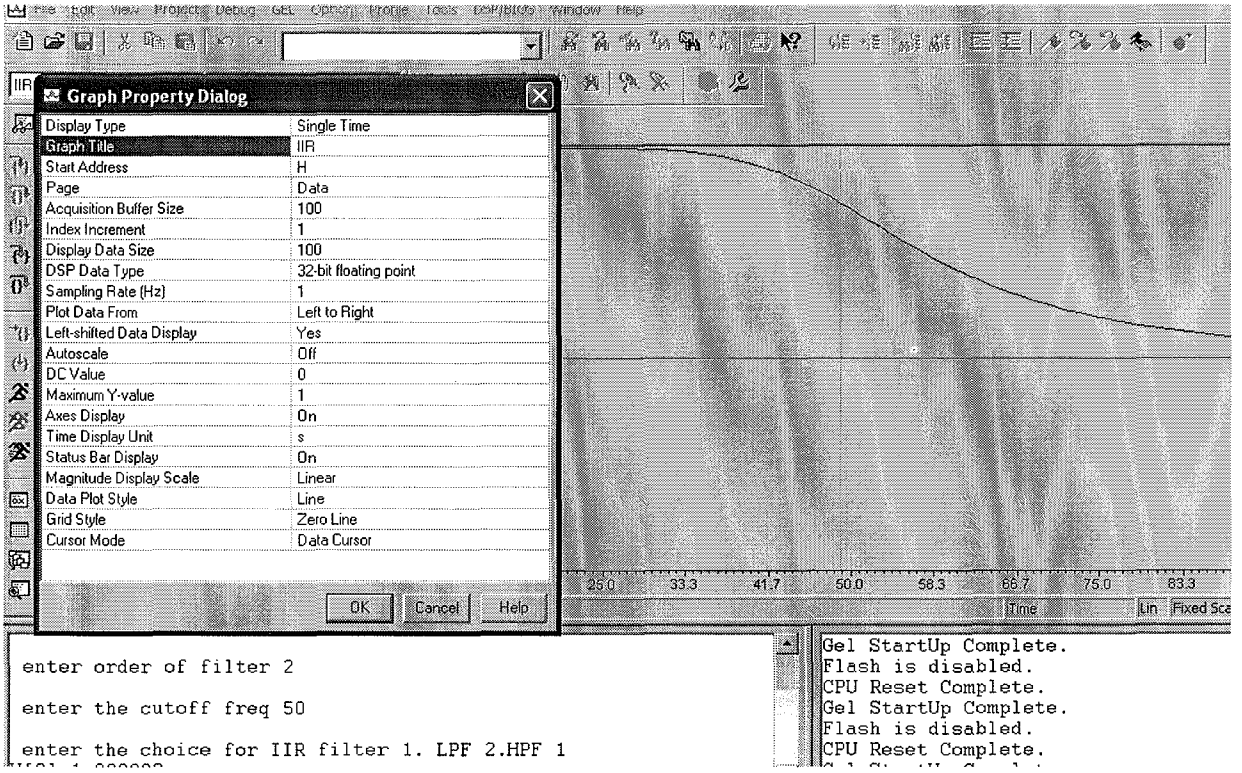
To design and implement IIR (LPF/HPF) filters using TMS320C5416 processor.

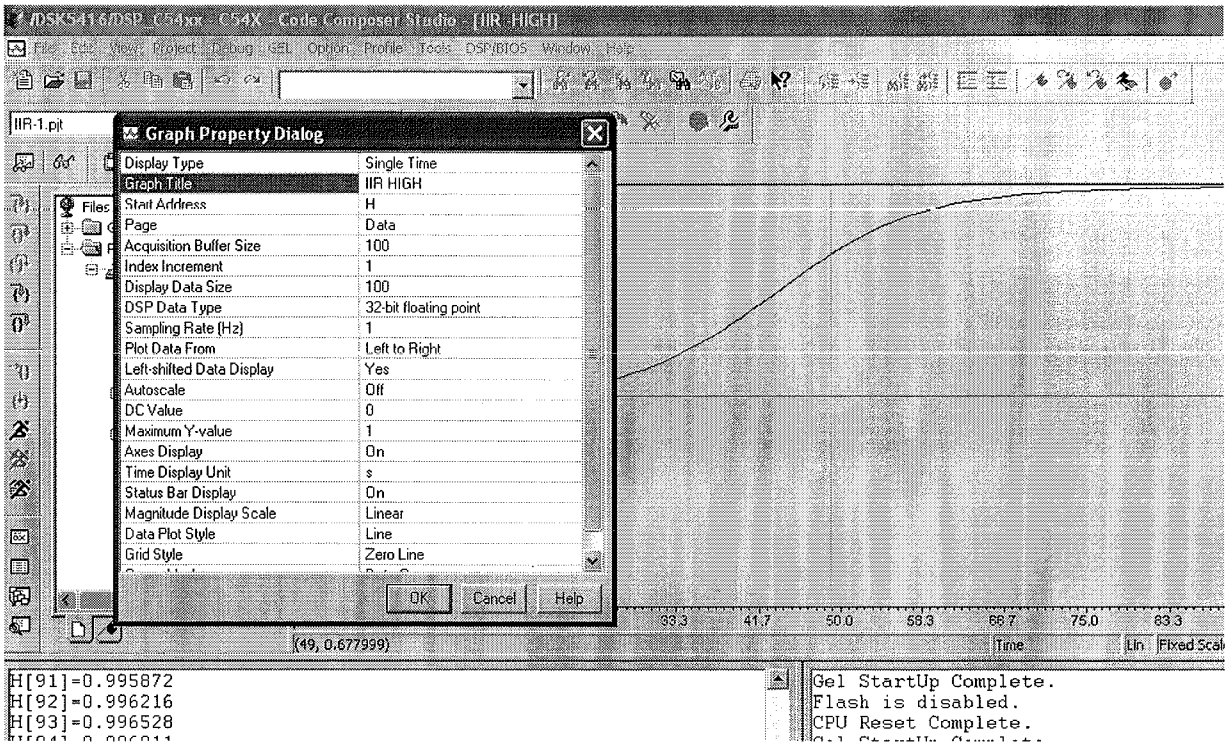
(A) IIR

PROGRAM:

```
#include<stdio.h>
#include<math.h>
int i,w,wc,c,N;
float H[100];
float mul(float, int);
void main()
{
        printf("\n enter order of filter ");
        scanf("%d",&N);
        printf("\n enter the cutoff freq ");
        scanf("%d",&wc);
        printf("\n enter the choice for IIR filter 1. LPF 2.HPF ");
        scanf("%d",&c);
        switch(c)
        {
        case 1:
                        for(w=0;w<100;w++)
                        {
                                H[w]=1/sqrt(1+mul((w/(float)wc),2*N));
                                printf("H[%d]=%f\n",w,H[w]);
                        }
                        break;
        case 2:
                        for(w=0;w<=100;w++)
                        {
                                H[w]=1/sqrt(1+mul((float)wc/w,2*N));
                                printf("H[%d]=%f\n",w,H[w]);
                        }
                        break;
        }
}
float mul(float a,int x)
{
```

```
        for(i=0;i<x-1;i++)
        a*=a;
        return(a);
}
```






```
H[91]=0.995872
H[92]=0.996216
H[93]=0.996528
```

```
Gel StartUp Complete.
Flash is disabled.
CPU Reset Complete.
```

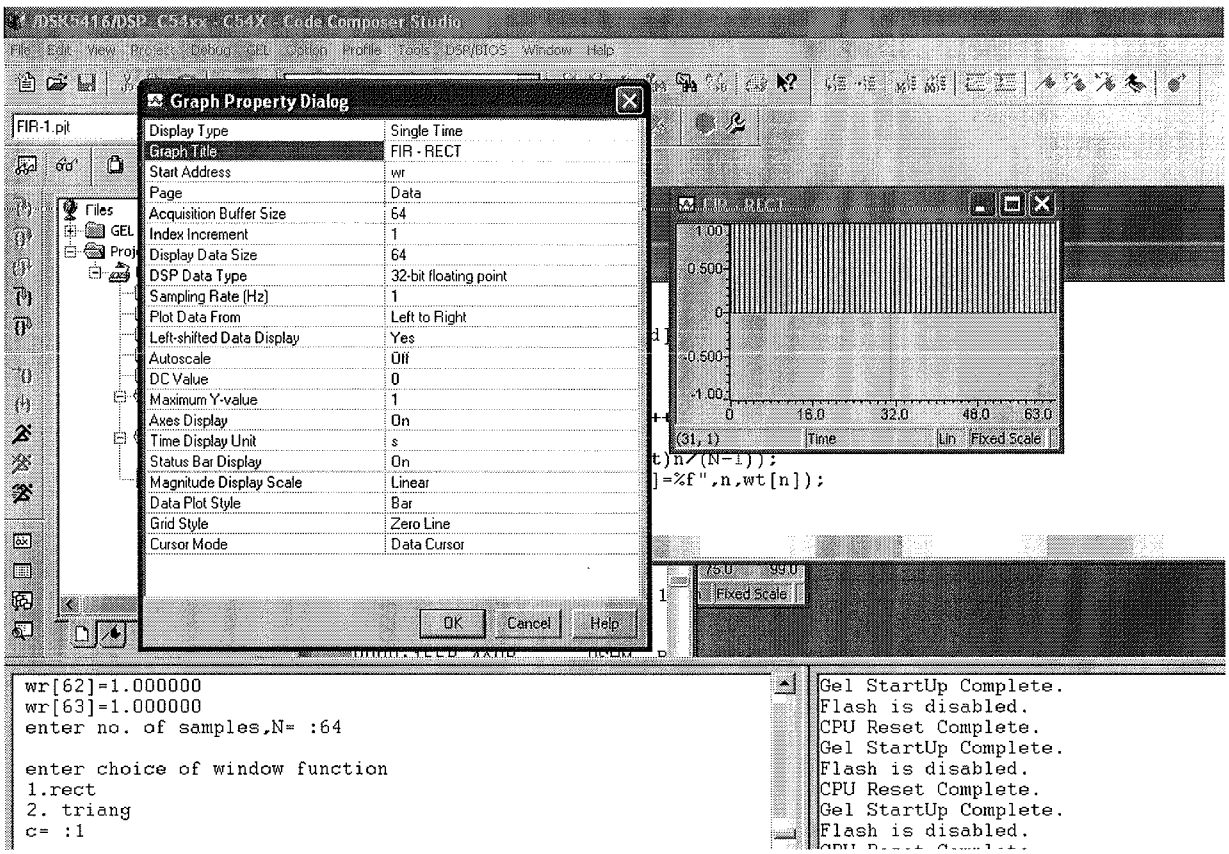
(B) FIR

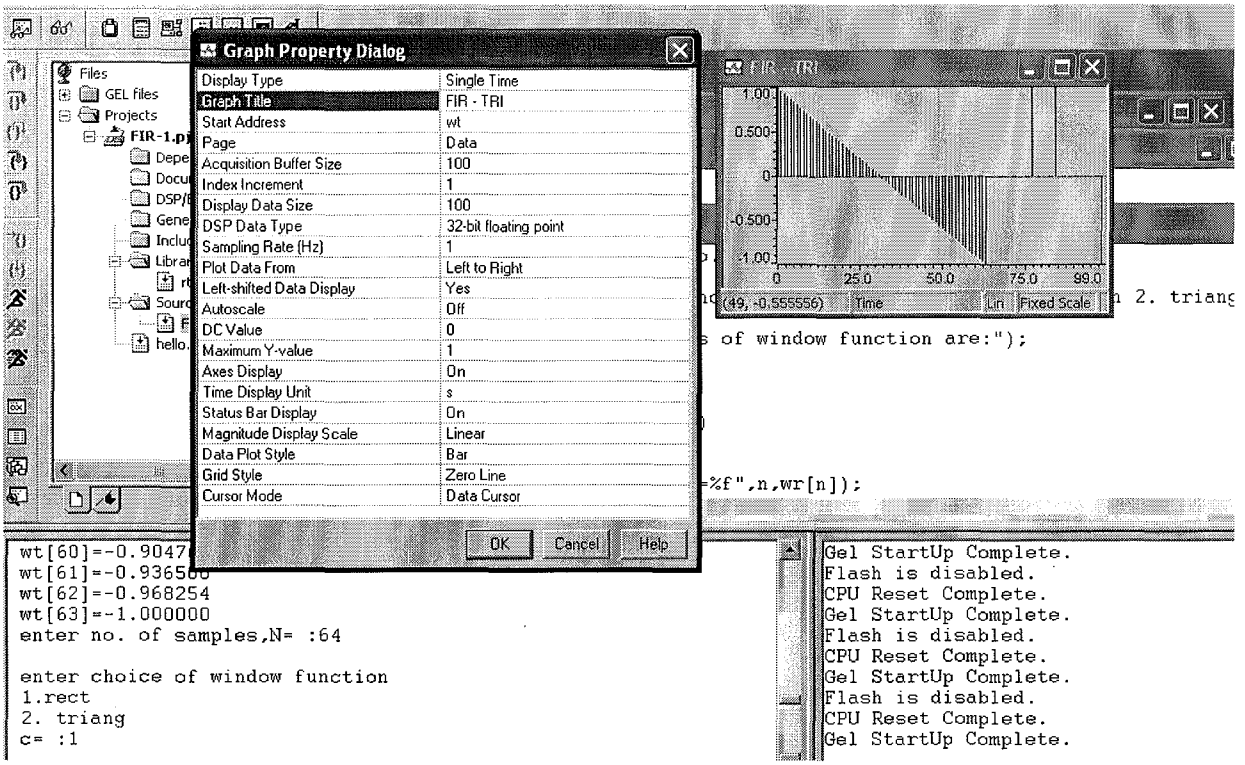
PROGRAM:

To verify FIR filters.

PROGRAM:

```
#include<stdio.h>
#include<math.h>
#define pi 3.1415
int n,N,c;
float wr[64],wt[64];
void main()
{
        printf("\n enter no. of samples,N= :");
        scanf("%d",&N);
        printf("\n enter choice of window function\n 1.rect \n 2. triang \n c= :");
        scanf("%d",&c);
        printf("\n elements of window function are:");
        switch(c)
        {
                case 1:
                        for(n=0;n<=N-1;n++)
                        {
                                wr[n]=1;
                                printf(" \n wr[%d]=%f",n,wr[n]);
                        }
                        break;
                case 2:
                        for(n=0;n<=N-1;n++)
                        {
                                wt[n]=1-(2*(float)n/(N-1));
                                printf("\n wt[%d]=%f",n,wt[n]);
                        }
                        break;
        }

}
```


Graph Property Dialog
Display Type
Single Time
Graph Title
FIR - RECT
Start Address
wr
Page
Data
Acquisition Buffer Size
64
Index Increment
1
Display Data Size
64
DSP Data Type
32-bit floating point
Sampling Rate (Hz)
1
Plot Data From
Left to Right
Left-shifted Data Display
Yes
Autoscale
Off
DC Value
0
Maximum Y-value
1
Axes Display
On
Time Display Unit
s
Status Bar Display
On
Magnitude Display Scale
Linear
Data Plot Style
Bar
Grid Style
Zero Line
Cursor Mode
Data Cursor
OK
Cancel
Help
FIR-1.pjt
Files
GEL
1.00
0.500
0
-0.500
-1.00
0
16.0
32.0
48.0
63.0
(31, 1)
Time
Lin
Fixed Scale
t)n/(N-1));
]=%f",n,wt[n]);
wr[62]=1.000000
wr[63]=1.000000
enter no. of samples,N= :64
enter choice of window function
1.rect
2. triang
c= :1
Gel StartUp Complete.
Flash is disabled.
CPU Reset Complete.
Gel StartUp Complete.
Flash is disabled.
CPU Reset Complete.
Gel StartUp Complete.
Flash is disabled.



RESULT:

Thus IIR and FIR filters were implemented using TMS320C5416 processor.

FFT IMPLEMENTATION USING TMS 320C5416

EXPERIMENT NO: 6 DATE:

AIM:

To design and implement FFT using TMS320C5416 processor.

PROGRAM:

```
#include<stdio.h>
#include<math.h>
#define SWAP(a,b); var=(a);(a)=(b);(b)=var;
void main()
{
int N,n,m,j,i,k,p;
float data[200],real1,imag1,real2,imag2,var;
float costheta,sinTheta,t,Theta;
printf("radix-2 ditfft algorithm\n\n");
printf("\nenter the no of samples in the sequence x(n),N=");
scanf("%d",&N);
printf("\nenter the samples of the sequence x(n)"
        "\n\nreal part imaginary part\n");
for(n=1;n<=N;n++)
{
printf("x(%d)=",n-1);
scanf("%f%f",&data[2*n-1],&data[2*n]);
}
n=N<<1;
j=1;
for(i=1;i<n;i=i+2)
{
if(j>i)
{
SWAP(data[j],data[i]);
SWAP(data[j+1],data[i+1]);
}
m=n>>1;

while(m>=2 && j>m)
{
j-=m;
m>>=1;
```

```
}
j+=m;
}
k=1;m=1;t=0.0;
while((N/(2*k))>=1)
{
p=pow(2,m);
n=1;
Theta=((2*3.14)/(float)p)*t;
costheta=cos(Theta);
sinTheta=sin(Theta);
for(i=1;i<=2*N;)
{
real1=data[i]+costheta*data[i+p]+sinTheta*data[i+1+p];
imag1=data[i+1]+costheta*data[i+1+p]-sinTheta*data[i+p];
real2=data[i]-costheta*data[i+p]-sinTheta*data[i+1+p];
imag2=data[i+1]-costheta*data[i+1+p]+sinTheta*data[i+p];
data[i]=real1;
data[i+1]=imag1;
data[i+p]=real2;
data[i+p+1]=imag2;
if(n<k)
{
t = t+1;
Theta = ((2*3.14)/(float)p)*t;
costheta = cos(Theta);
sinTheta = sin(Theta);
}
else
{
i = i+p+2;
n = 1;
t =0;
Theta = ((2*3.14)/(float)p)*t;
costheta = cos(Theta);
sinTheta = sin(Theta);
}
}
k = k<<1;
m++;
}

printf("\n Output of DIT FFT is as follows\n");
printf("\n Real Part of X(k)      Imaginary Part of X(k)");

for(n=1;n<=N;n++)
```

```
{
printf("\n %f\t\t %f",data[2*n-1],data[2*n]);
}
}
```

RESULT:

Thus the FFT were implemented using TMS320C5416 processor.

ADDITIONAL PROGRAMS

SAMPLING & EFFECT OF ALIASING

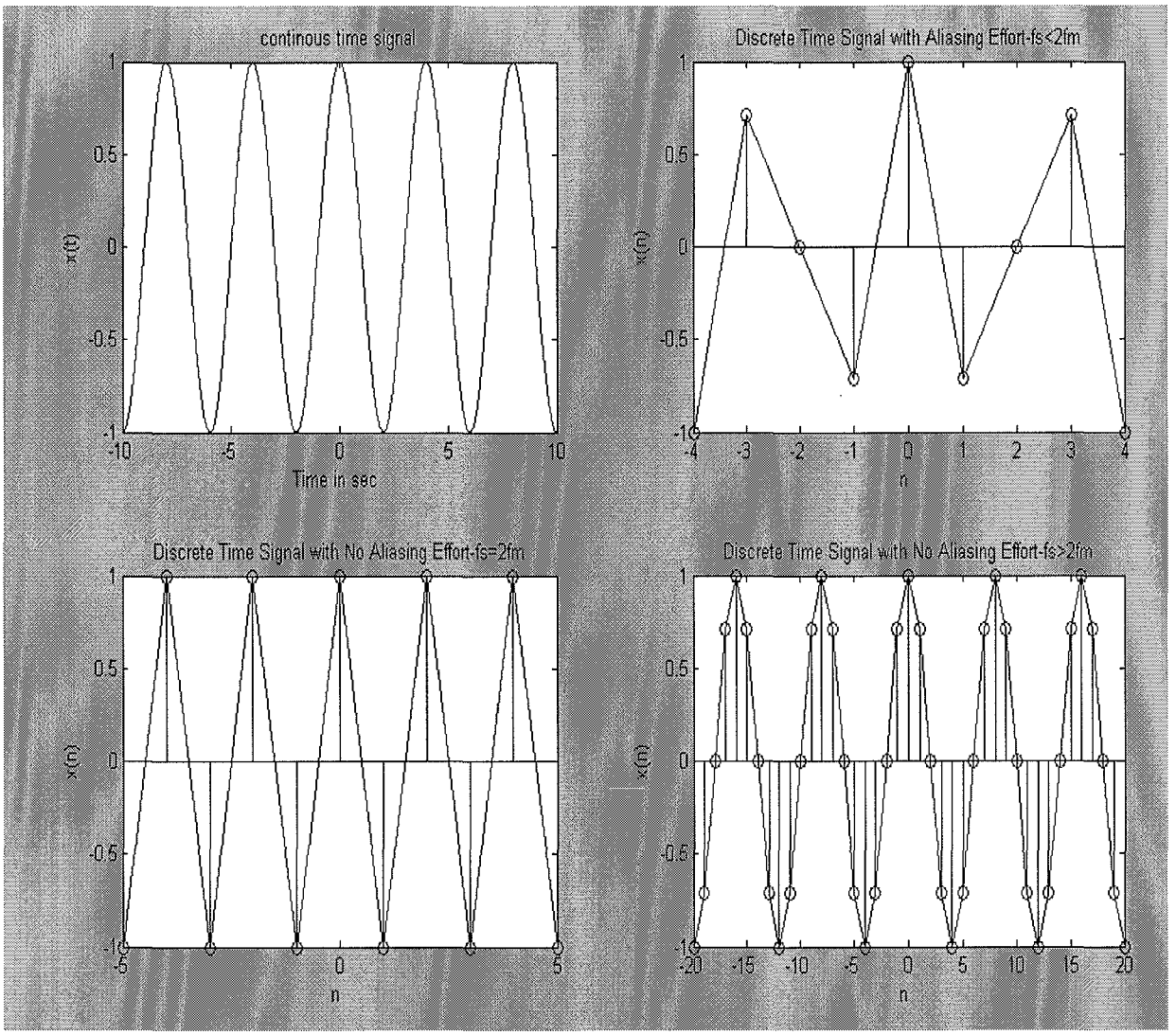
EXPERIMENT NO: 1 DATE:

AIM:

To Perform Sampling of Continuous time Signal with different sampling frequency in order to study aliasing effect.

PROGRAM:

```
clc;clear all; close all;
t=-10: .01:10;
T=4;fm=1/T;
x=cos(2*pi*fm*t) ;
fs1=1.6*fm;fs2=2*fm;fs3=8*fm;

n1=-4:1:4;
xn1=cos(2*pi*n1*fm/fs1) ;
subplot(2,2,1) ;plot (t,x) ;
xlabel('Time in sec');ylabel('x(t) ');title('Continous time signal');

subplot(2,2,2) ;stem(n1,xn1) ;
hold on
subplot(2,2,2) ;plot(n1,xn1);
xlabel ('n');ylabel ( 'x (n) ' ) ;
title('Discrete time signal with aliasing effect-fs<2fm');
n2=-5:1:5;xn2=cos(2*pi*n2*fm/fs2) ;

subplot(2,2,3) ;stem(n2,xn2) ;
hold on
subplot(2,2,3) ;plot (n2,xn2) ;
xlabel ('n');ylabel ( 'x (n) ' );
title('Discrete time signal with no aliasing effect-fs=2fm');
n3=-20:1:20;xn3=cos(2*pi*n3*fm/fs3) ;

subplot(2,2,4) ;stem(n3,xn3);
hold on
subplot(2,2,4) ;plot(n3,xn3);
xlabel ('n');ylabel ( 'x (n) ' ) ;
title('Discrete time signal with no aliasing effect-fs>2fm');
```



RESULT:

Thus the Sampling of Continuous time Signal with different sampling frequency was performed.

CALCULATION OF FFT OF A SIGNAL

EXPERIMENT NO: 2 DATE:

AIM:

To compute Discrete Fourier Transform of the input sequence using FFT Function.

FORMULA:

$$y(k) = \sum_{n=0}^{N-1} x(n)e \quad for\ 0 \leq k \leq N-1$$

PROGRAM:

```
clc; close all; clear all;
x = input ('Enter the Sequence :');
n = input ('Enter the length of fft:') ;
y= fft(x,n) ;subplot(3,1,1); stem(y) ;
ylabel('Imaginary Value--->') ;
xlabel('Integer 'K' Value-->') ;
y1= abs(y);subplot(3,1,2) ;stem(y1) ;
ylabel('Magnitude--->') ;
xlabel ('k-->')
y2= real(y) ;y3=imag(y);y4=atan(y3./y2); subplot(3,1,3) ; stem(y4) ;
ylabel('Phase--->') ;
xlabel('k-->')
disp('The FFT Sequence is: ') ;y,y1,y2,y3,y4
```

Enter the Sequence : [0 1 2 3 4 5 6 7]
Enter the length of fft :8
The FFT Sequence is:

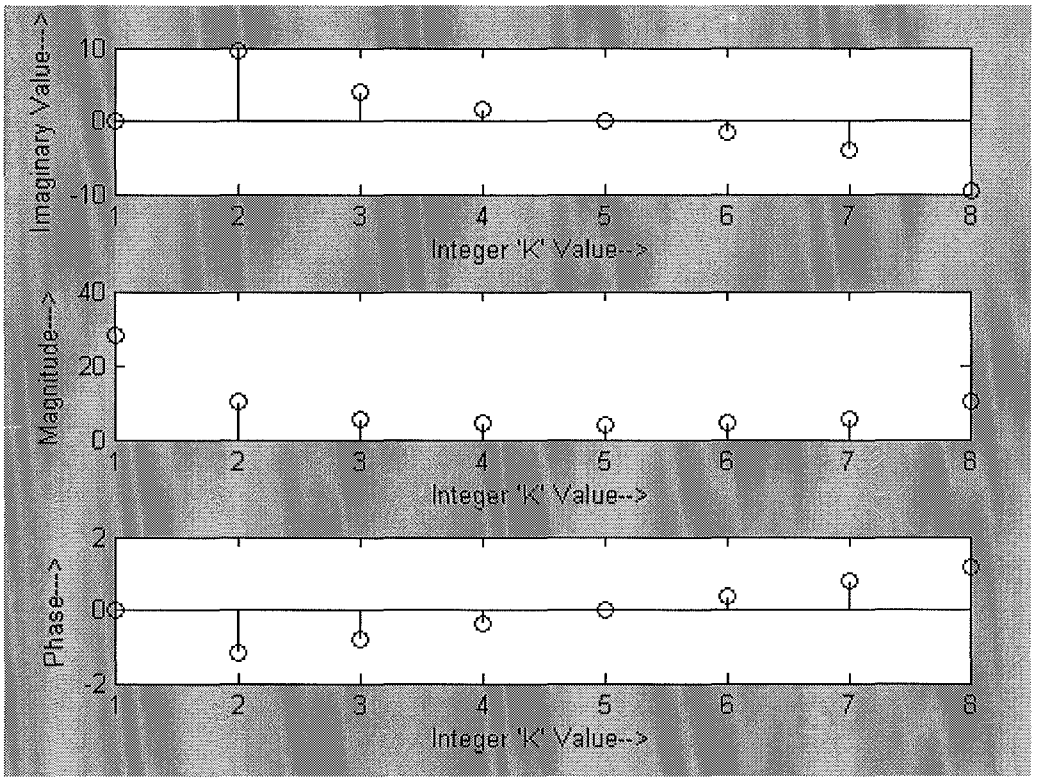
y = 28.0000 -4.0000 + 9.6569i -4.0000 + 4.0000i -4.0000 + 1.6569i -4.0000 -4.0000 - 1.6569i -4.0000 - 4.0000i -4.0000 - 9.6569i

y1 = 28.0000 10.4525 5.6569 4.3296 4.0000 4.3296 5.6569 10.4525

y2 = 28 -4 -4 -4 -4 -4 -4 -4

y3 = 0 9.6569 4.0000 1.6569 0 -1.6569 -4.0000 -9.6569

y4 = 0 -1.1781 -0.7854 -0.3927 0 0.3927 0.7854 1.1781



RESULT:

Thus the Discrete Fourier Transform of the input Sequence is computed using FFT function.

DOWN SAMPLING AND UP SAMPLING

EXPERIMENT NO: 3 DATE:

AIM:

To implement Downsampling and Upsampling operation using MATLAB.

PROGRAM:

```
clc;
clear all;
close all;
x = input('Enter the input Signal,x=');
ni = input('Enter the Starting index,ni=');
M = input('Enter the downsampling factor,M=');
len = length(x);
if(ni<=0)
   n = [ni:ni+len-1];
else
   n = [0:ni+len-1];
   x = [zeros(1,ni),x];
end

m = fix(n(1)/M):fix(n(end)/M);
a = 1;
y = zeros(1,length(m));
for i = min(m):max(m)
   y(a) = x(n==M*i);
   a = a+1;
end
subplot(211);
stem(n,x);
title('x[n]');
xlabel('n');
subplot(212);
stem(m,y);
title('y[m]=x[M*n]');
xlabel('m');
```

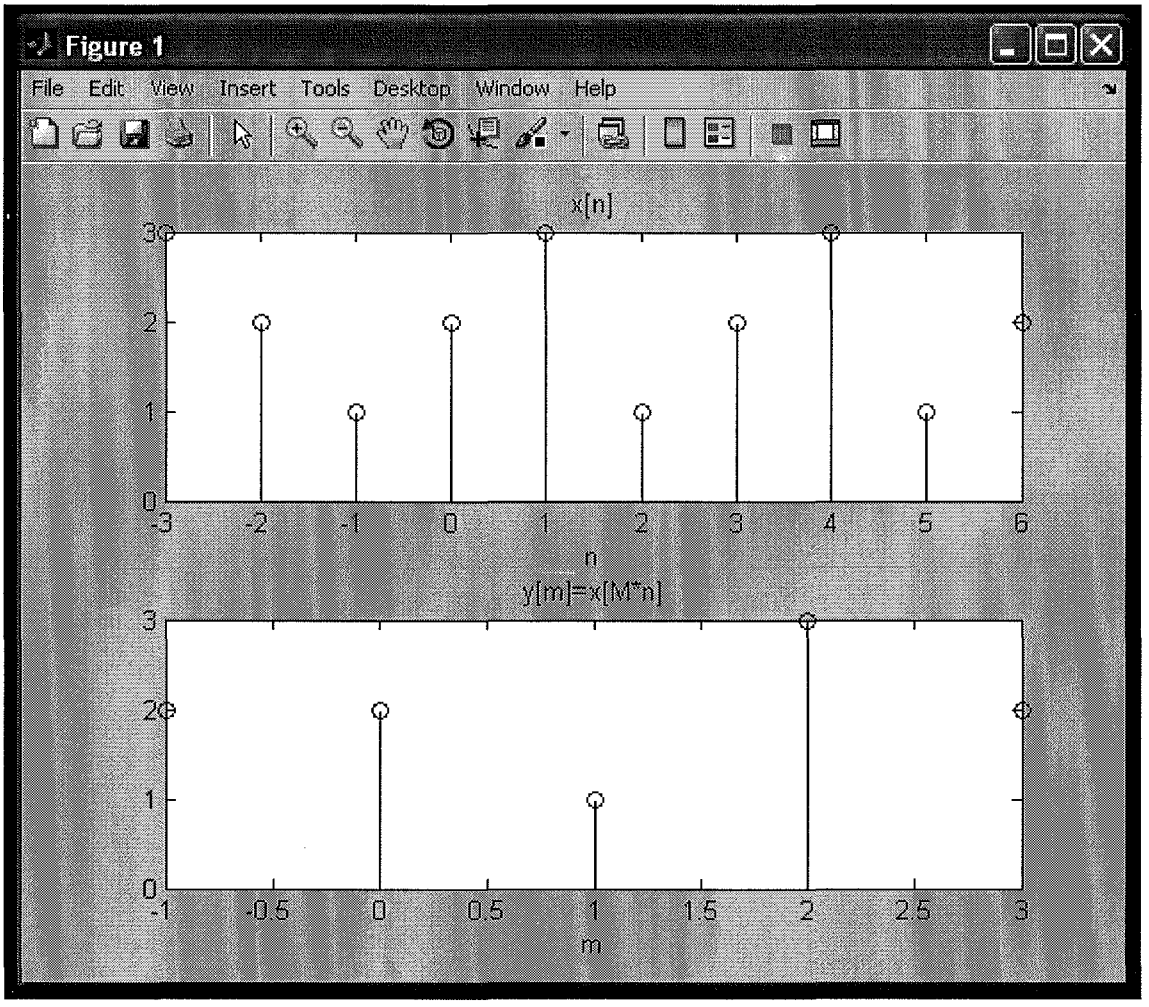
As an Example:

Enter the input Signal,x=[3 2 1 2 3 1 2 3 1 2]

Enter the Starting index,ni=3

Enter the downsampling factor,M=2

DOWN SAMPLING



PROGRAM:

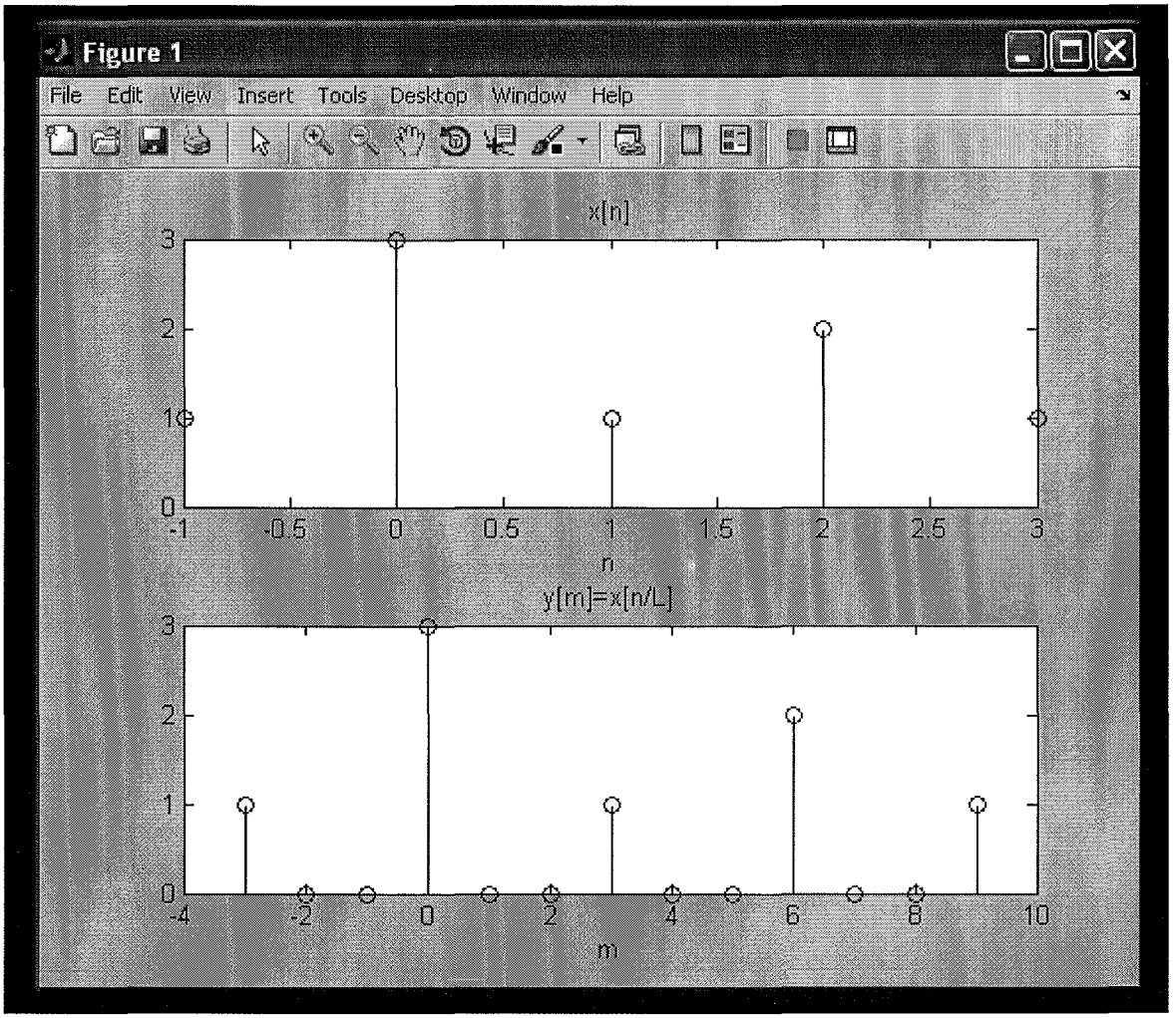
```
clc;
clear all;
close all;
x = input('Enter the input Signal,x=');
ni = input('Enter the Starting index,ni=');
L = input('Enter the upsampling factor,L=');
len = length(x);
if(ni<=0)
   n = [ni:ni+len-1];
else
   n = [0:ni+len-1];
   x = [zeros(1,ni),x];
end

m = fix(n(1)*L):fix(n(end)*L);
a = 1;
y = zeros(1,length(m));
for i = min(m):max(m)
   if (i/L==fix(i/L))
      y(a) = x(n==i/L);
   end
   a = a+1;
end
subplot(211);
stem(n,x);
title('x[n]');
xlabel('n');
subplot(212);
stem(m,y);
title('y[m]=x[n/L]');
xlabel('m');
```

As an Example:

Enter the input Signal,x=[1 3 1 2 1]

Enter the Starting index,ni=-1

Enter the upsampling factor,L=3



RESULT:

Thus the Down sampling and Up sampling operation was implemented using MATLAB.

WAVEFORM GENERATION

EXPERIMENT NO: 4 DATE:

AIM:

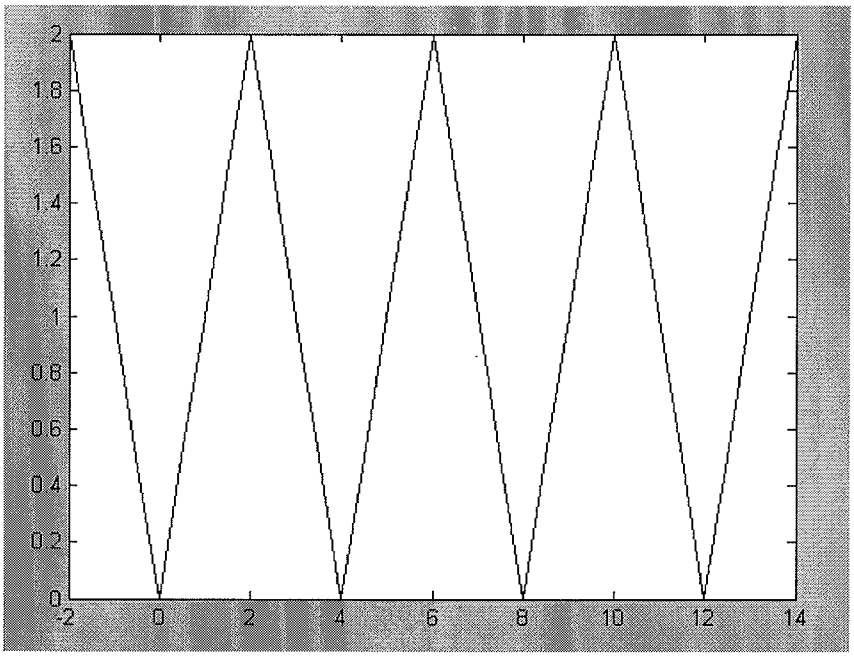
To Generate Triangular, Sawtooth, and Square waveform using MATLAB Program.

PROGRAM:

a) TRIANGULAR WAVEFORM

```
y= 0:0.5:2;
for j = 0:3
   x = ( 4*j) + y;
   plot(x,y);
   hold on
end
for k = 0:3
   x = ( 4*k)  - y;
   plot(x,y);
   hold on
end
hold off
```

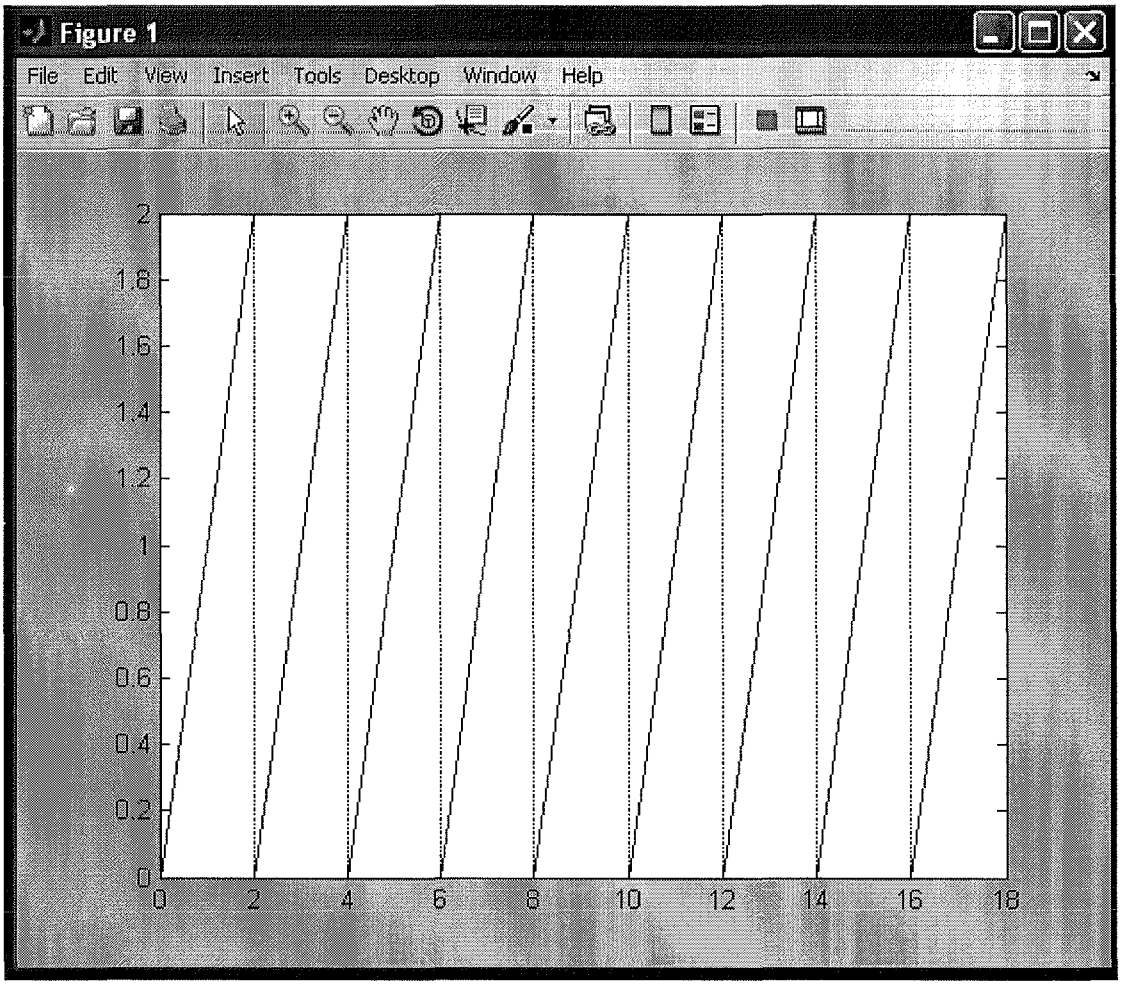
TRIANGULAR WAVEFORM



b) SAWTOOTH WAVEFORM

```
y = 0:.5: 2;
for j = 0:8
   A = ( 2*j) + y;
   plot (A,y, 'r');
   hold on
end
x=2:2:18
for k = 0:.01:2
   b=k;
   plot(x,b,'b');
   hold on
end
hold off
```

SAWTOOTH WAVEFORM

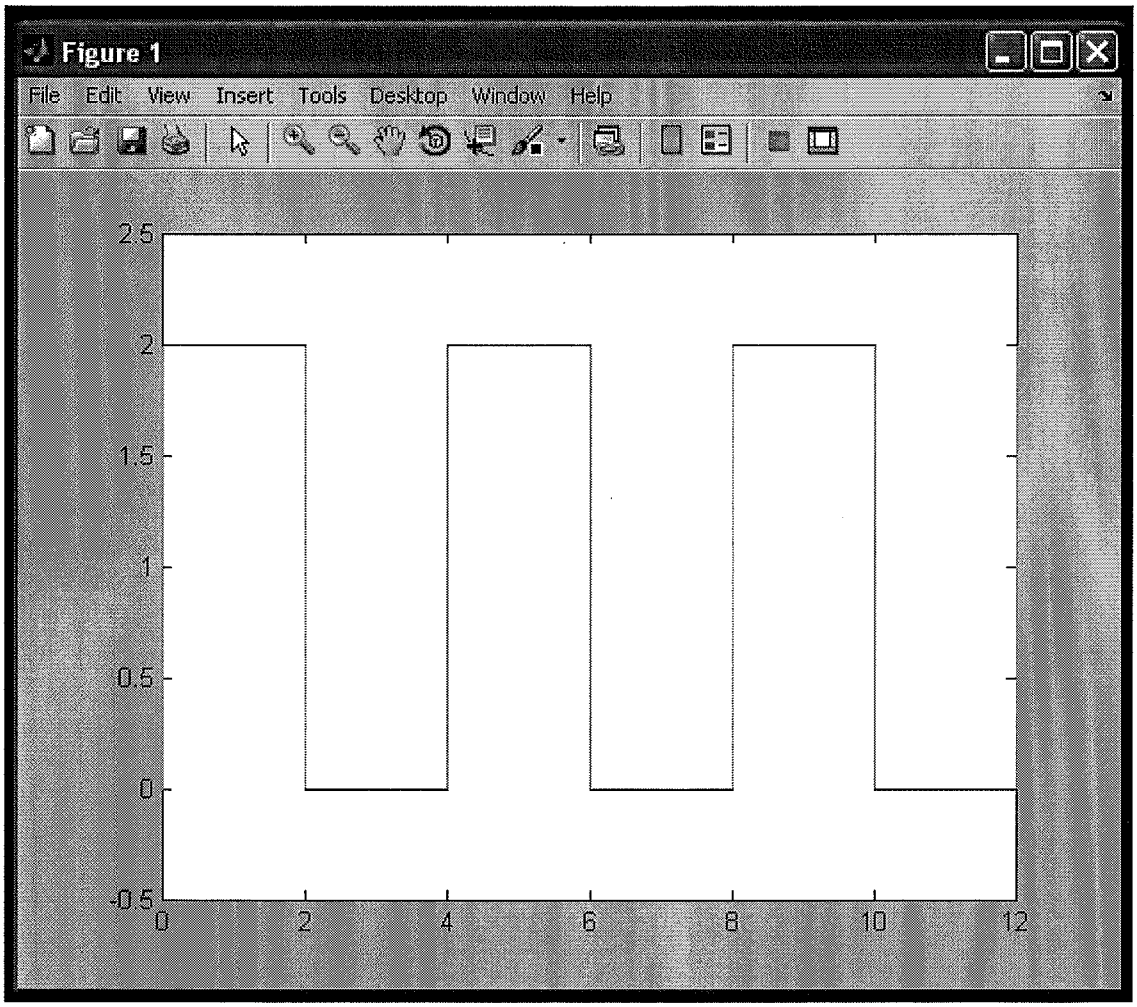


c) - SQUARE WAVEFORM

```
Y= 0: .001:2;
for j = 0:2:12;
x = y;
plot(j,x,'r');
hold on
end
for k = 0:4:12
x= k+y;
end
for k = 2:4:12
X= k+y;
end
hold off
m 2;
```

m 0;
axis([O 12 -0.5 2.5])
RESULT:
plot(x,m, 'r'); hold on
plot(x,m, 'r'); hold on

SQUARE WAVEFORM



RESULT:

Thus the Triangular, Sawtooth and Square waveform was generated using MA TLAB Program.

SUM OF SINUSOIDAL SIGNALS

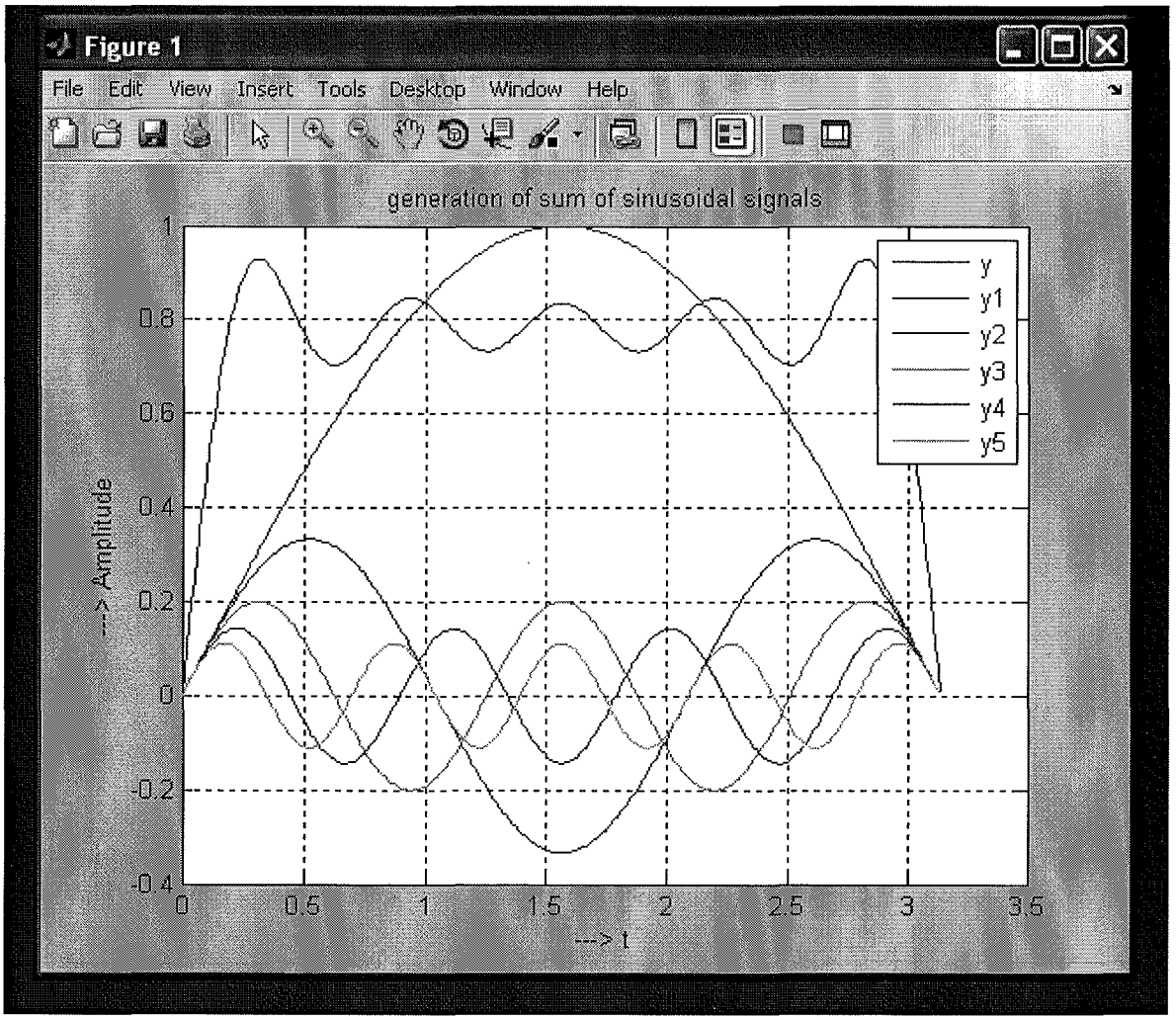
EXPERIMENT NO: 5 DATE:

AIM:

To verify Sum of Sinusoidal Signals using MATLAB

PROGRAM:

```
                                % sum of sinusoidal signals
clc;clear all;close all;
tic;
                                   %giving linear spaces
t=0:.01:pi;
% t=linspace(0,pi,20);
                                %generation of sine signals
y1=sin(t);
y2=sin(3*t)/3;
y3=sin(5*t)/5;
y4=sin(7*t)/7;
y5=sin(9*t)/9;
y = sin(t) + sin(3*t)/3 + sin(5*t)/5 + sin(7*t)/7 + sin(9*t)/9;
plot(t,y,t,y1,t,y2,t,y3,t,y4,t,y5);
legend('y','y1','y2','y3','y4','y5');
title('generation of sum of sinusoidal signals');grid;
ylabel('---> Amplitude');xlabel('---> t');
toc;
```



RESULT:

Thus the Sum of Sinusoidal Signals was generated using MA TLAB Program.

WAVE GENERATION

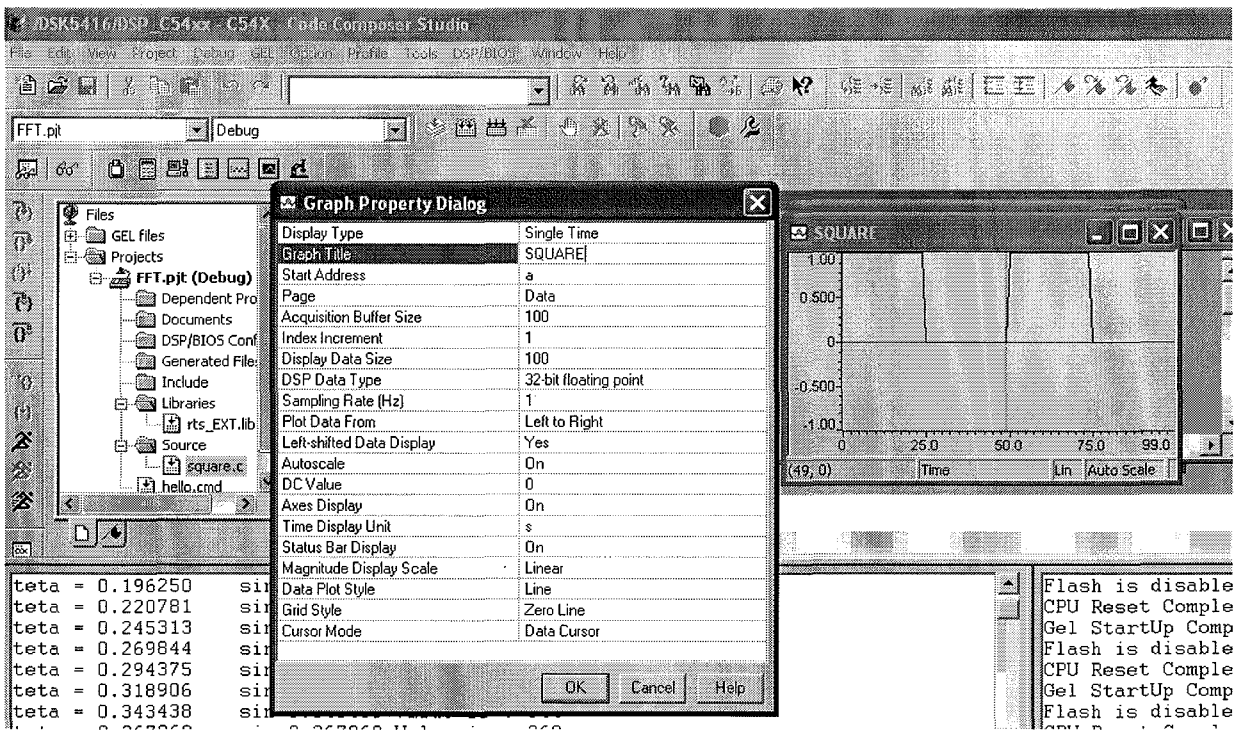
EXPERIMENT NO: 6 DATE:

AIM:

To Generate a Square waveform using TMS320C5416 DSP KIT

Program

```
#include <stdio.h>
#include <math.h>
unsigned short a[500];
void main( )
{
int i=0,j=0,k=0;
        for(k=0;k<5;k++)
{
for(i=0;i<50;i++)
{
a[j] = 0x0000FFFF;
 j++;
}
for(i=0;i<50;i++)
        {
                a[j] = 0x0; j++;
                printf("%f",a[j]);
        }
}
}
```

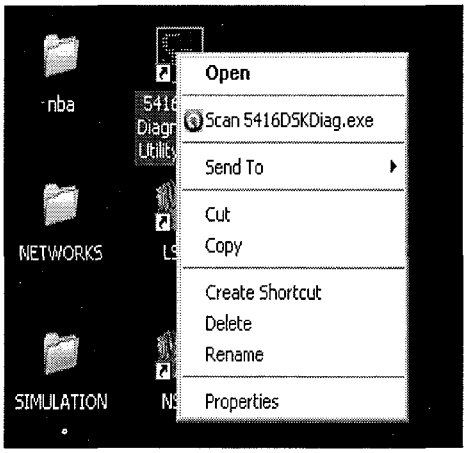


RESULT

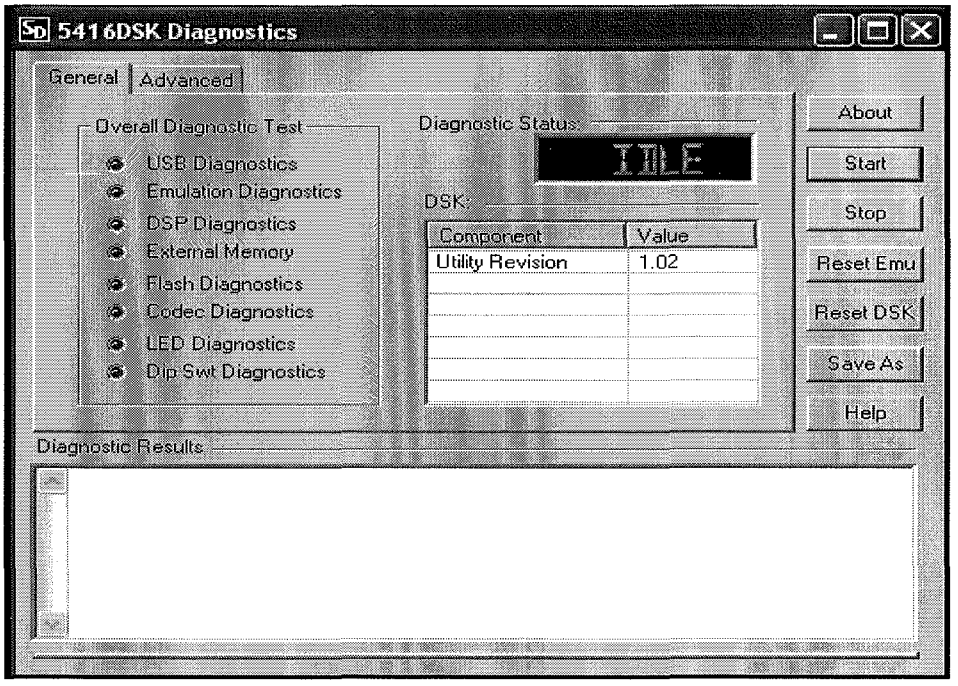
Thus the Square waveform was generated.

PROCEDURE TO WORK WITH CODE COMPOSER STUDIO V3.1

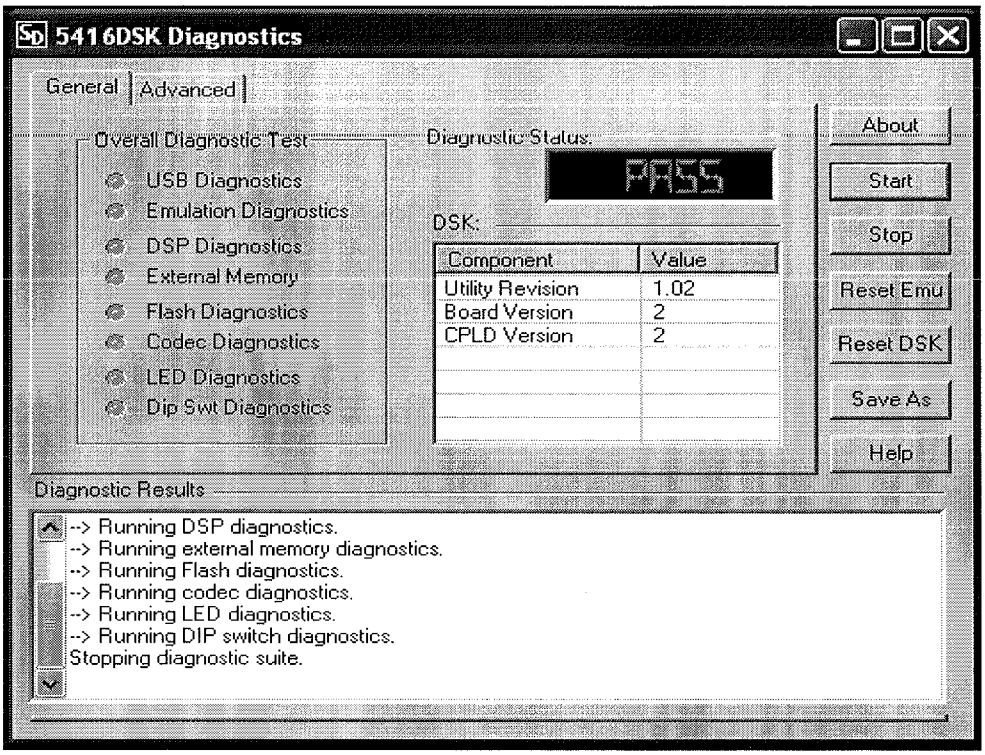
STEP 1: Check the working condition of Processor – TMS320C5416.



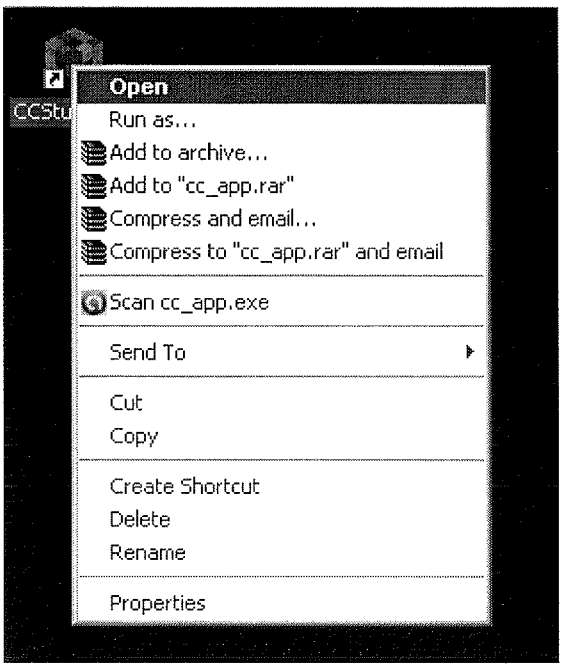
STEP 2: Click start



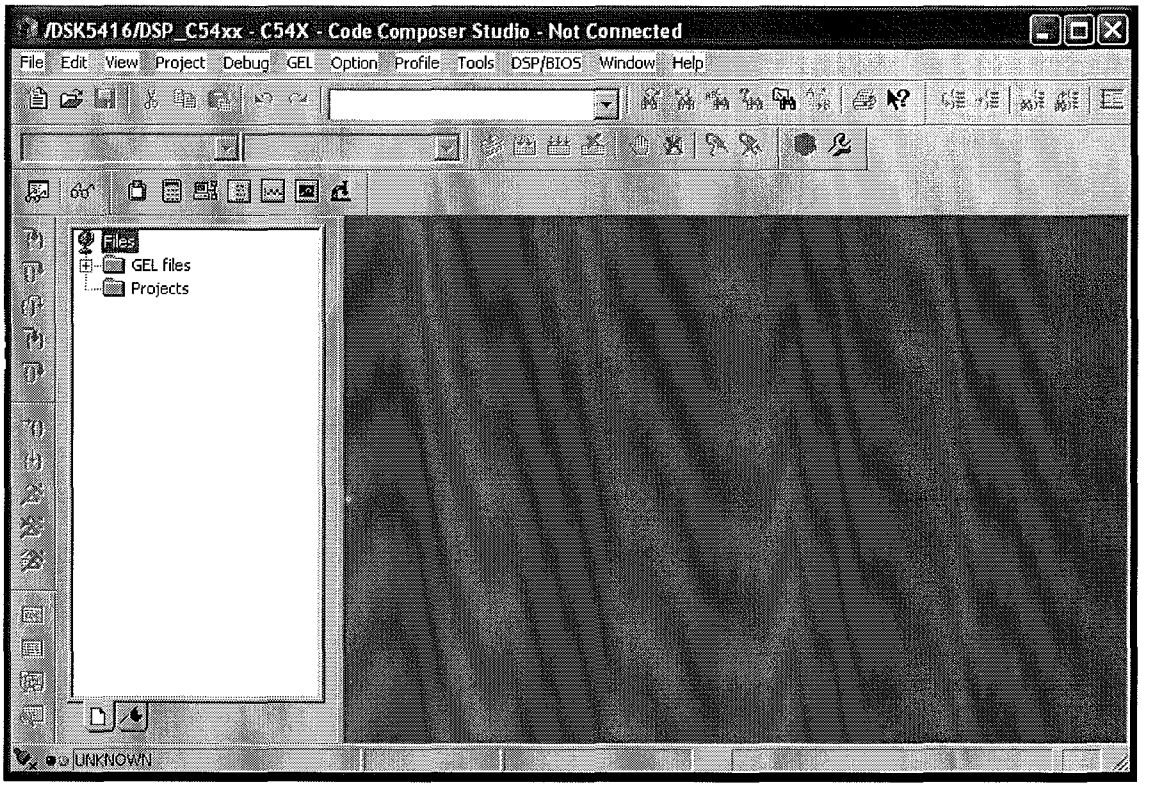
STEP 3: The 'PASS' display ensures the proper working condition of Processor.



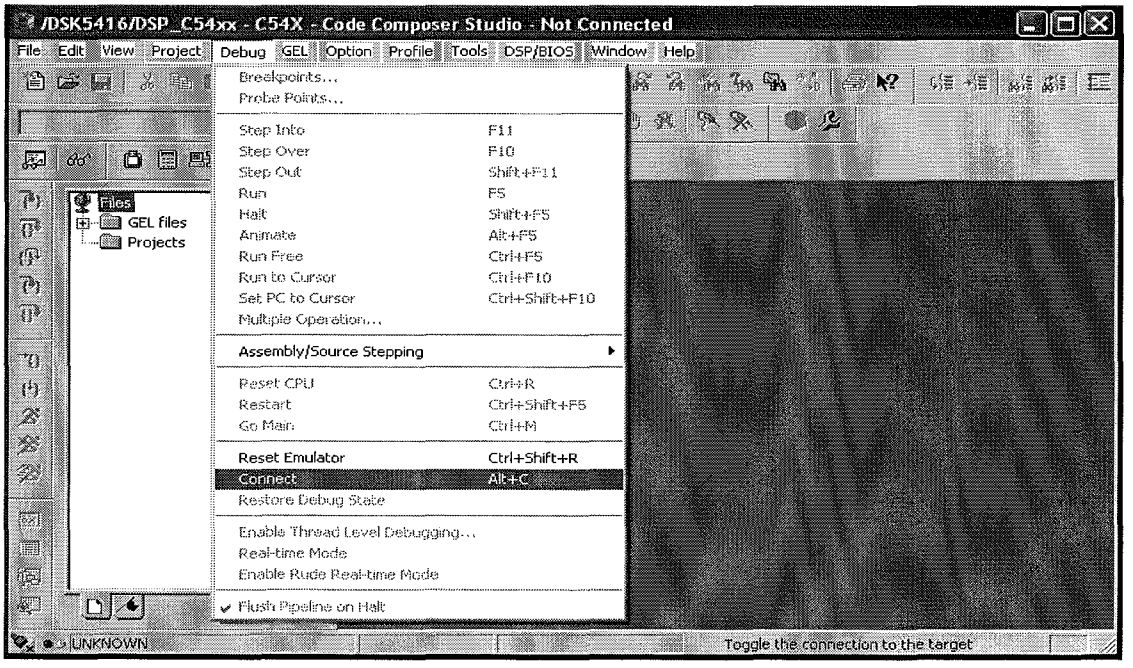
STEP 4: Open CCStudio icon displayed in the desktop.



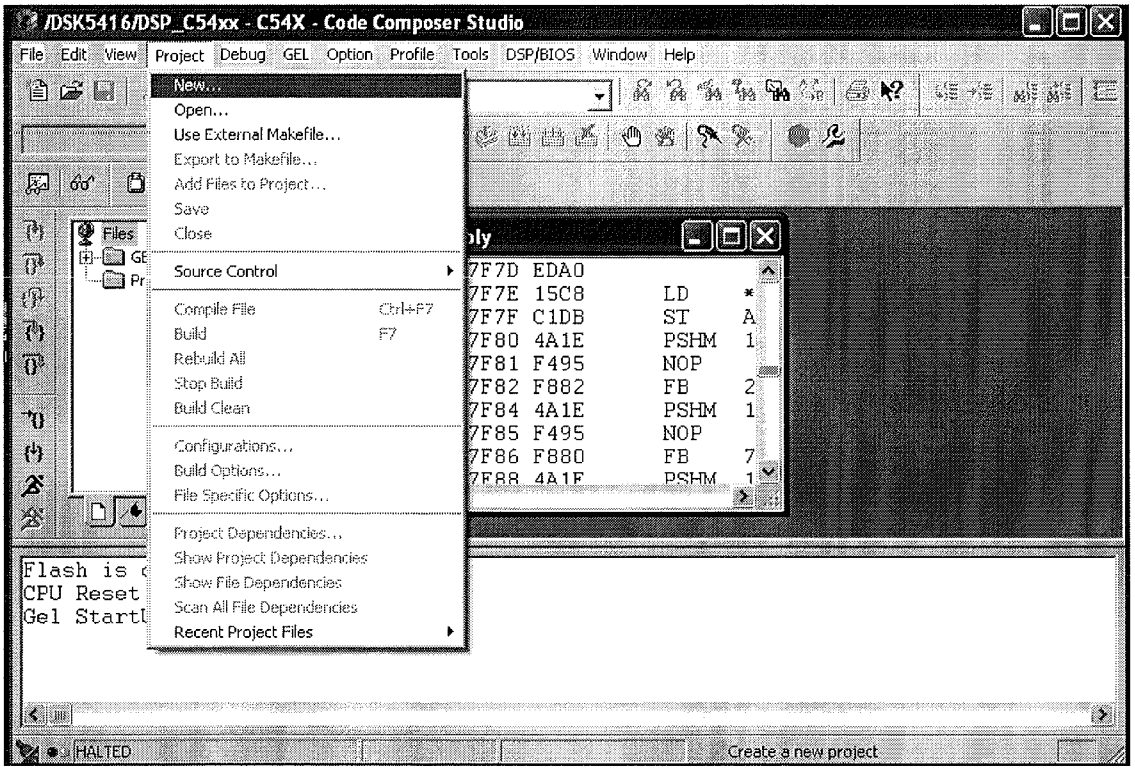
STEP 5: Initially 'Not connected' status is displayed.



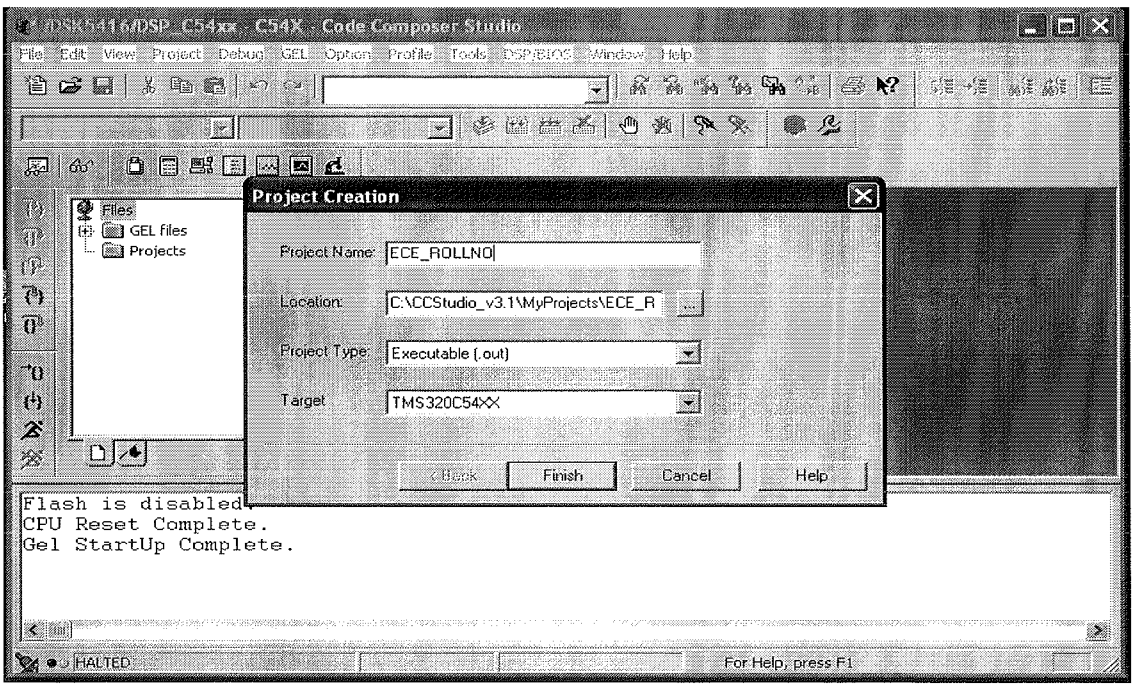
STEP 6: Click 'Debug' and select 'Connect' from the drop down list.



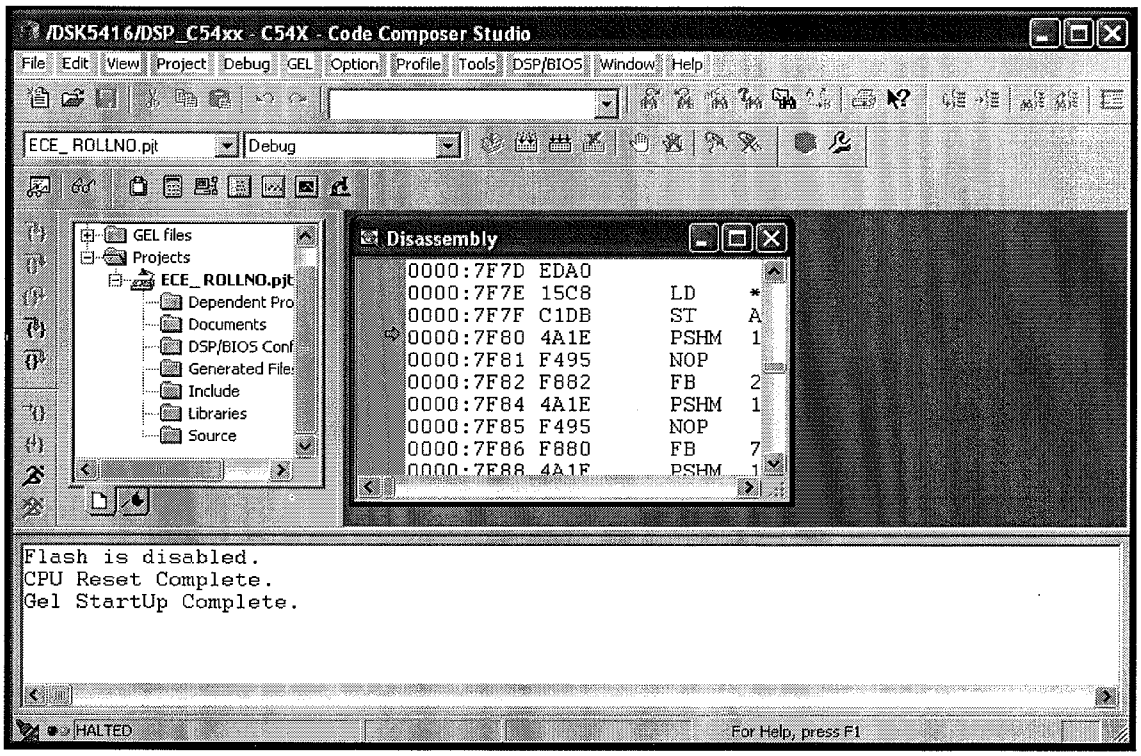
STEP 7: Click 'Project' and select 'New'



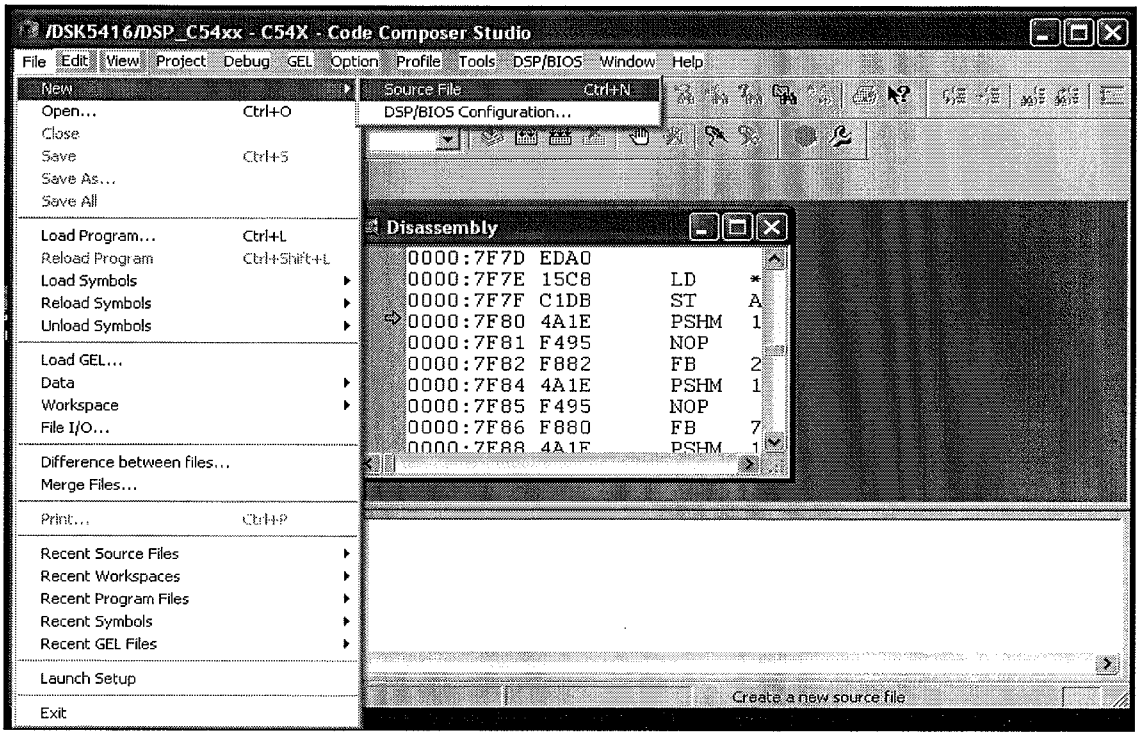
STEP 8: Enter the project name and click 'Finish'



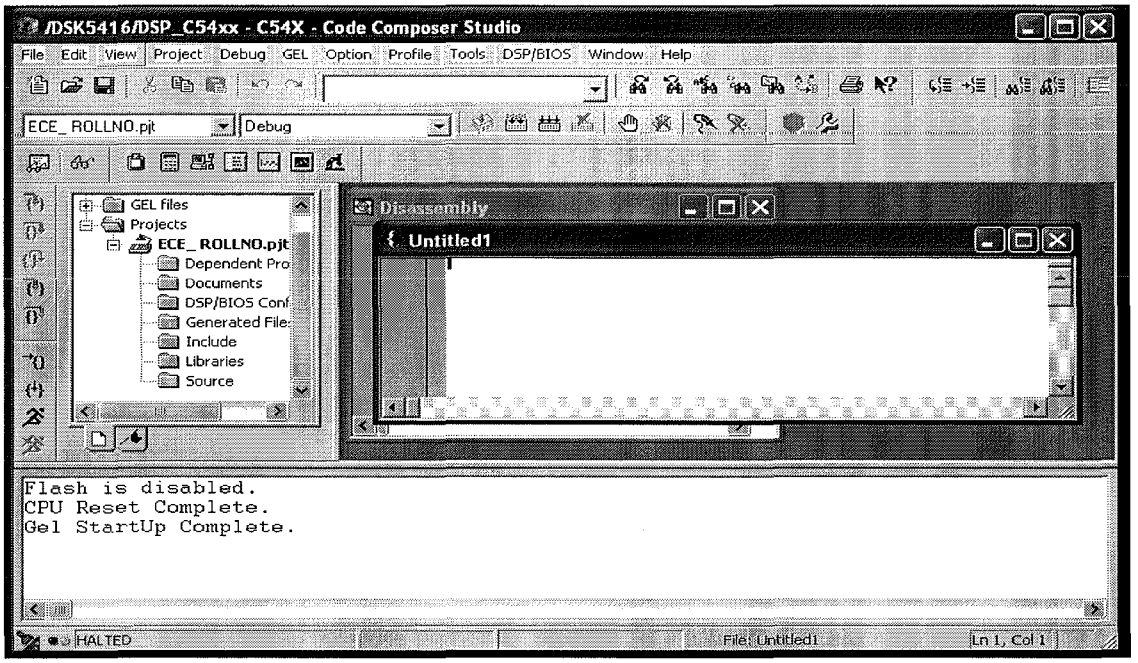
STEP 9: The selected project will be displayed with the extension .pjt



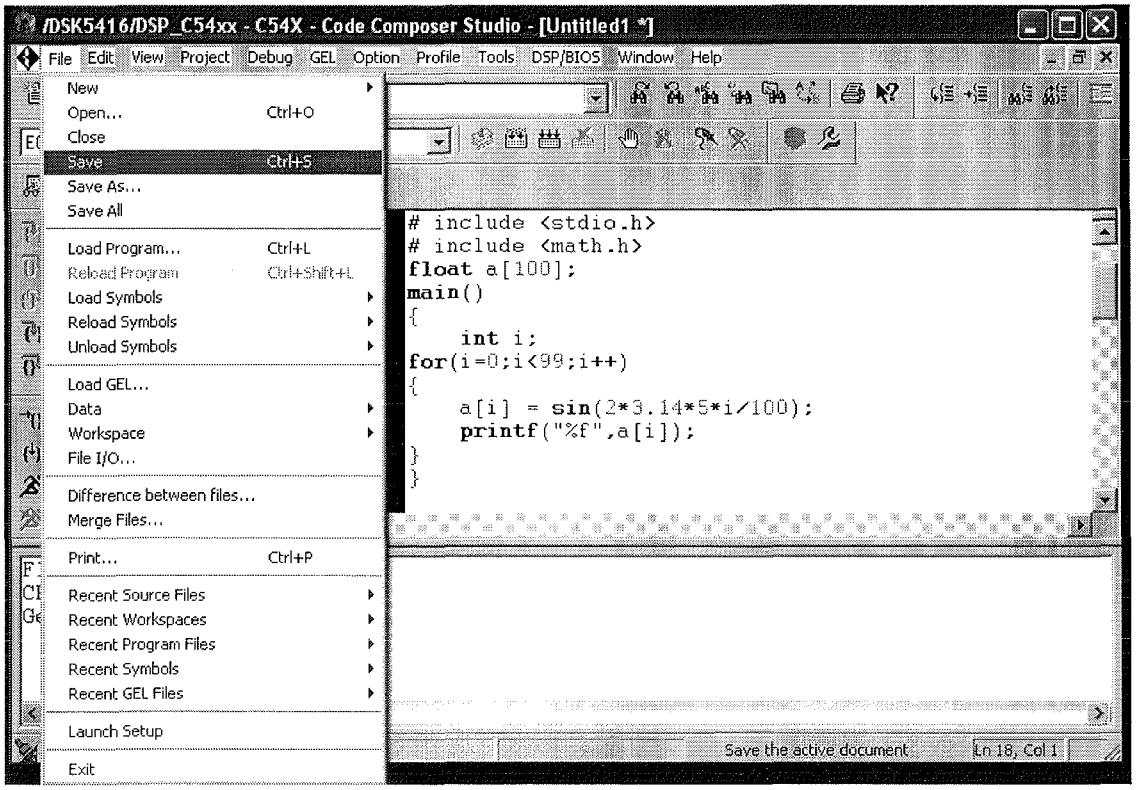
STEP 10: Go to File – New – Source file



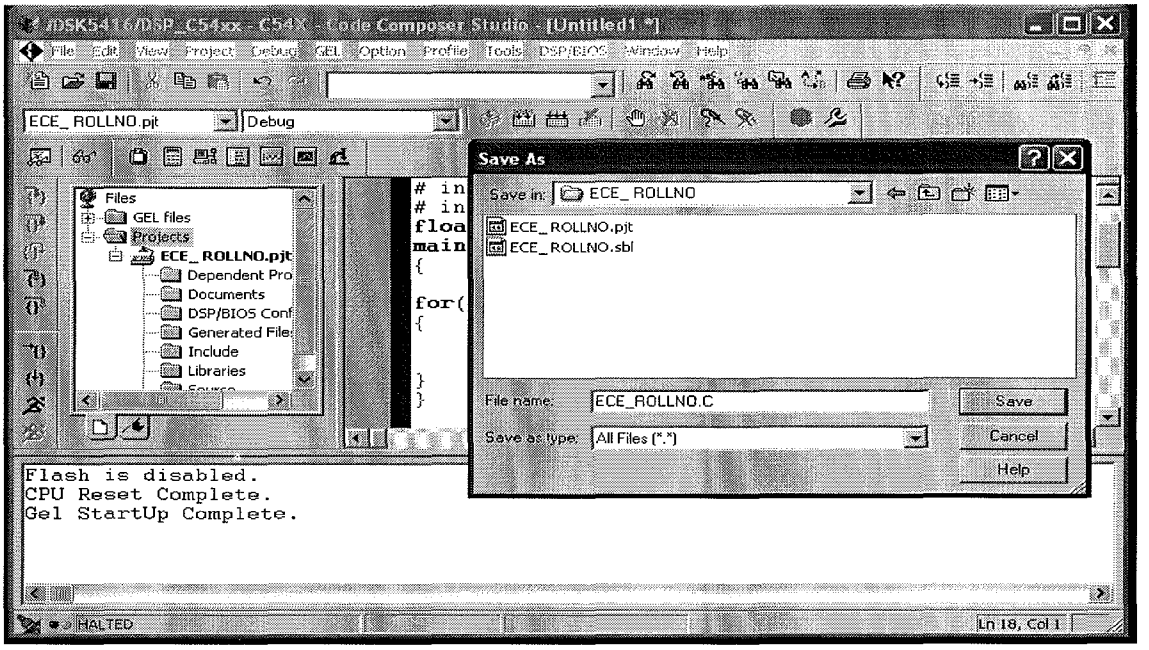
STEP 11 : Enter the coding in the untitled window.



STEP 12: Click File – Save

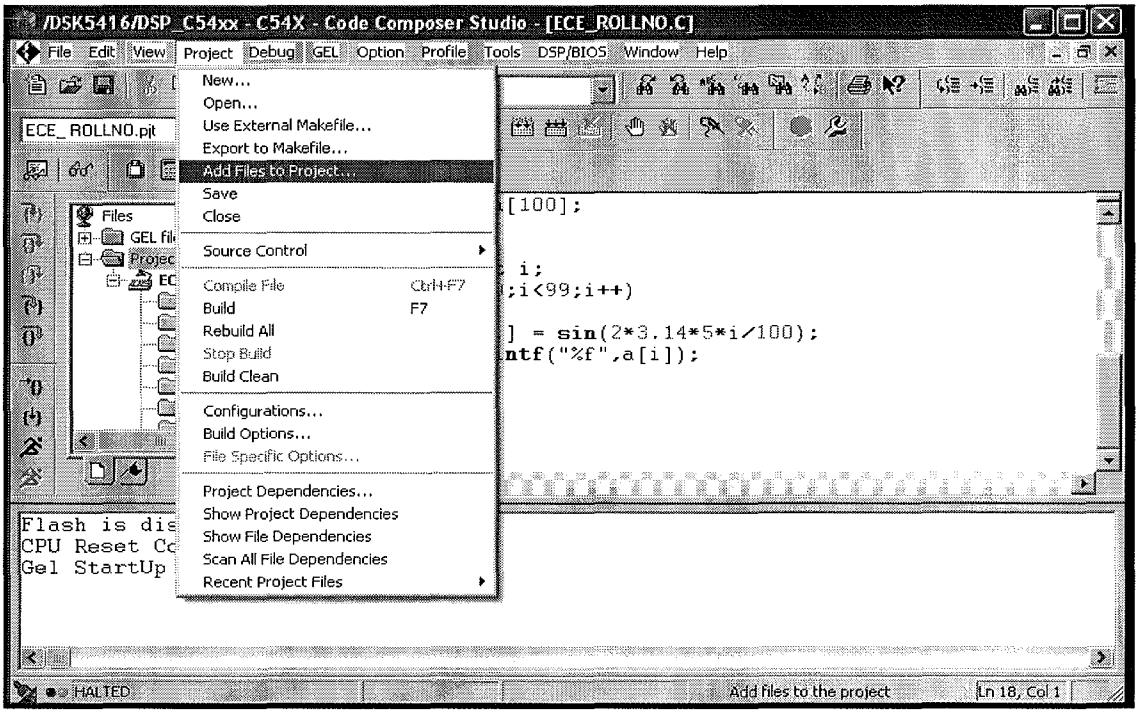


STEP 13: Save the file in your project with extension .c (File name and project name should be the same)

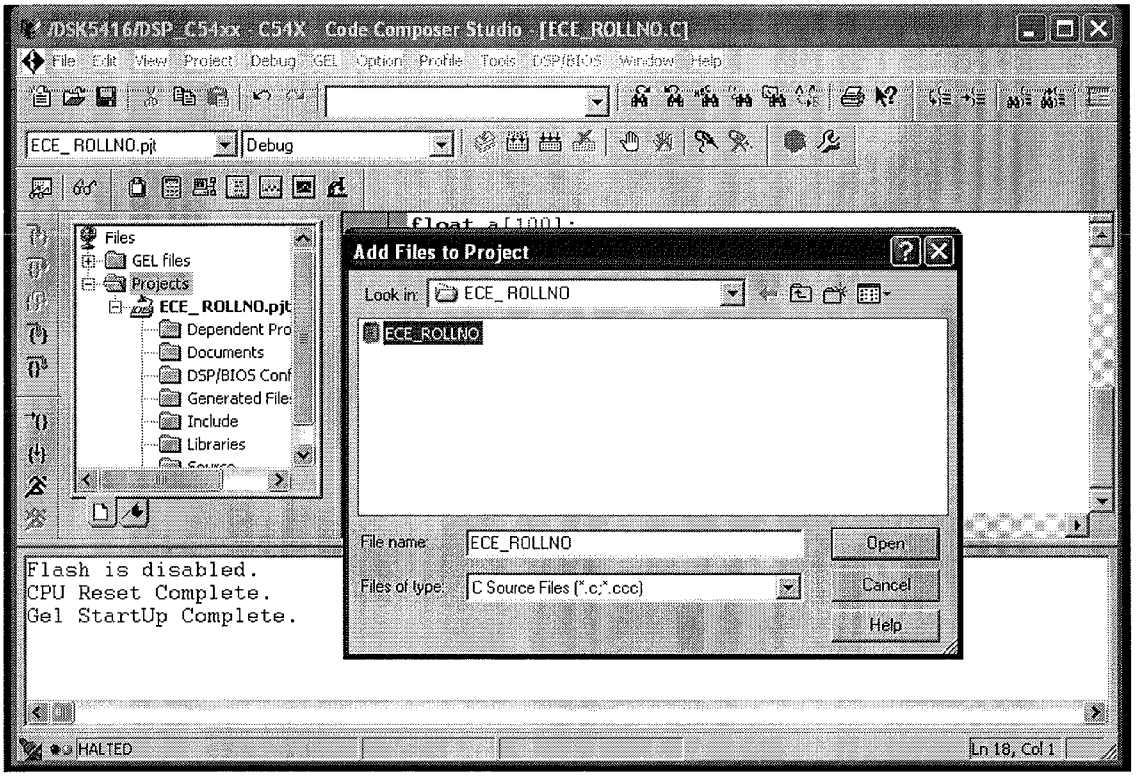


STEP 14: Before executing the coding, 3 files (Source file (.c file), Library file (rts_ext) & Command file (Hello)) has to be added to the project.

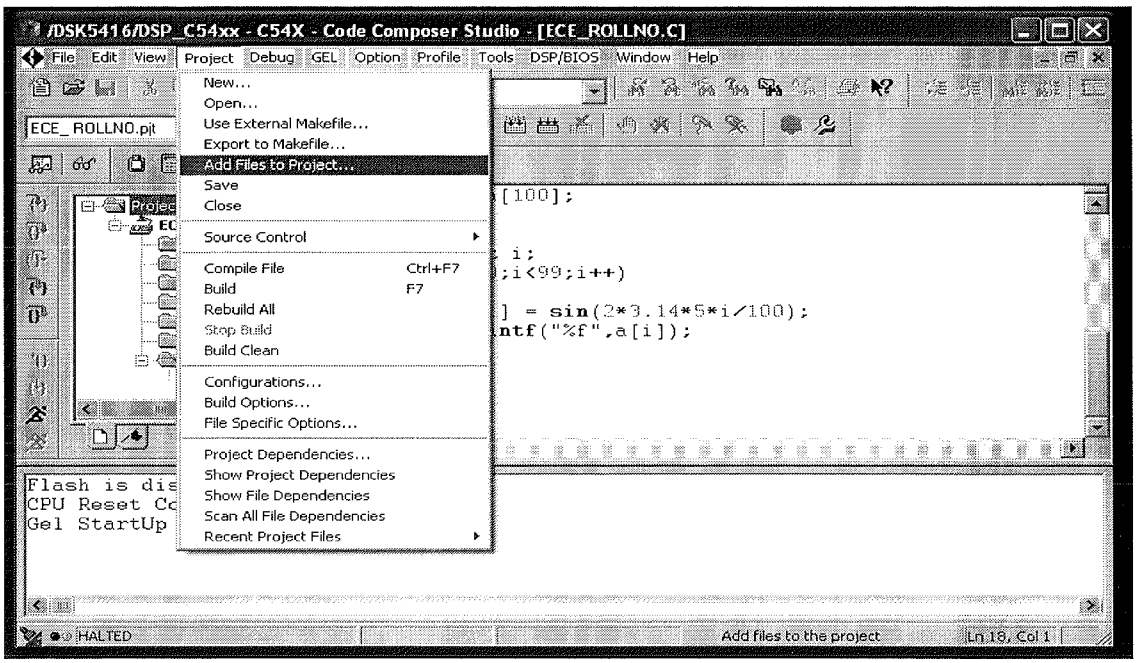
Go to Project – Add files to project



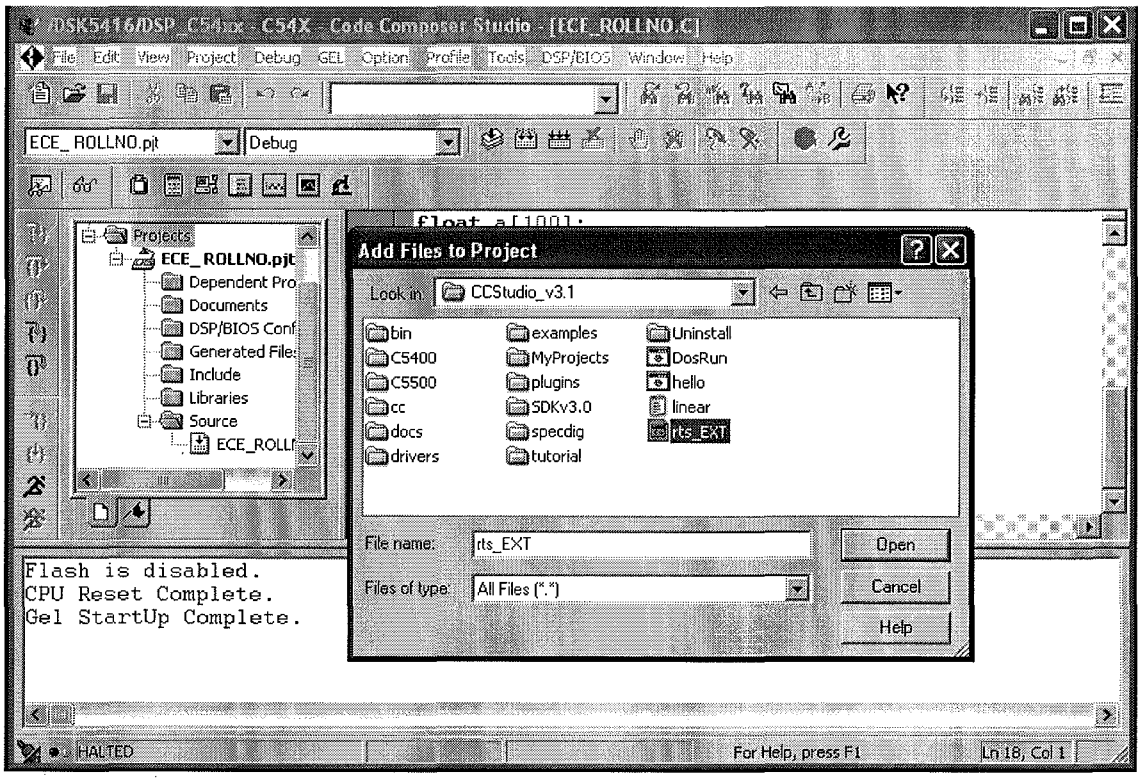
STEP 15: If the file is properly saved, file will be available in the project itself. Click that C file.



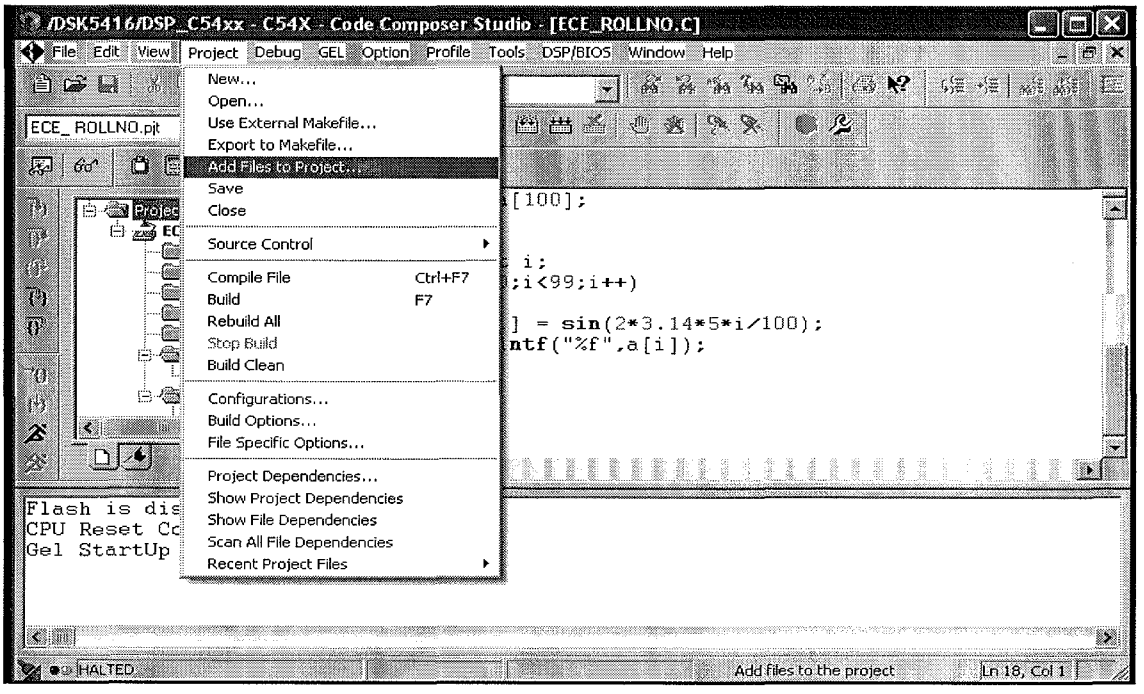
STEP 16: Go to Project – Add files to project.



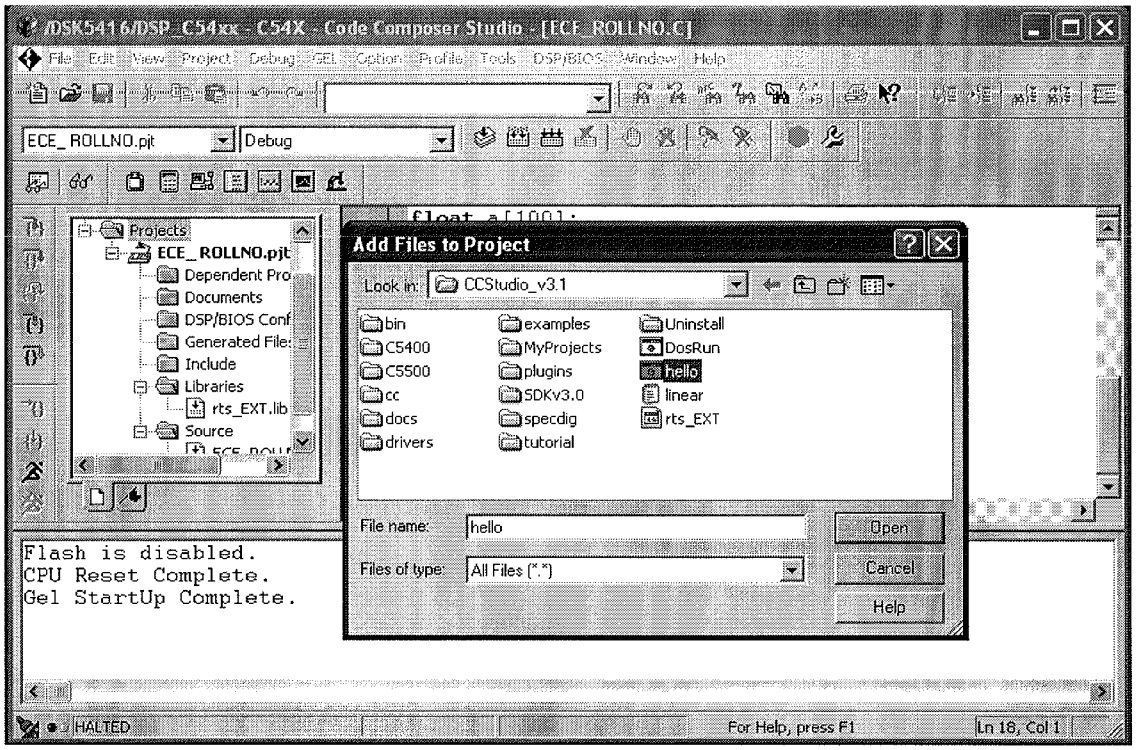
STEP 17: Look in ' CCStudio_v3.1'. Select 'All files' in Files of type and select file 'rts_EXT'.



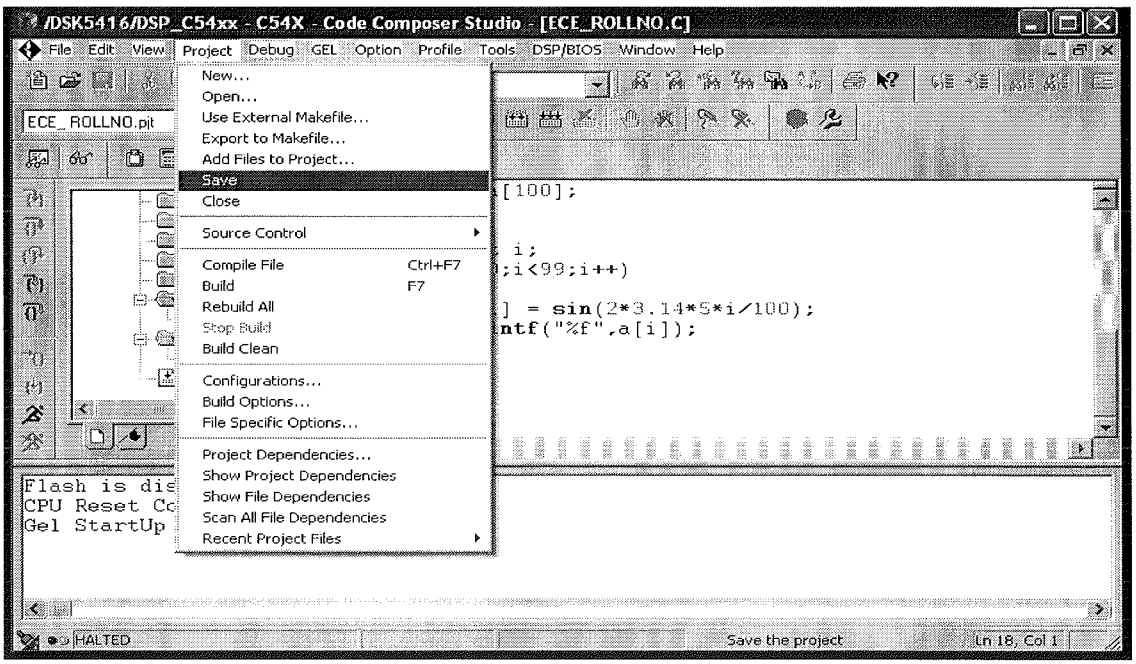
STEP 18: Go to Project – Add files to project.



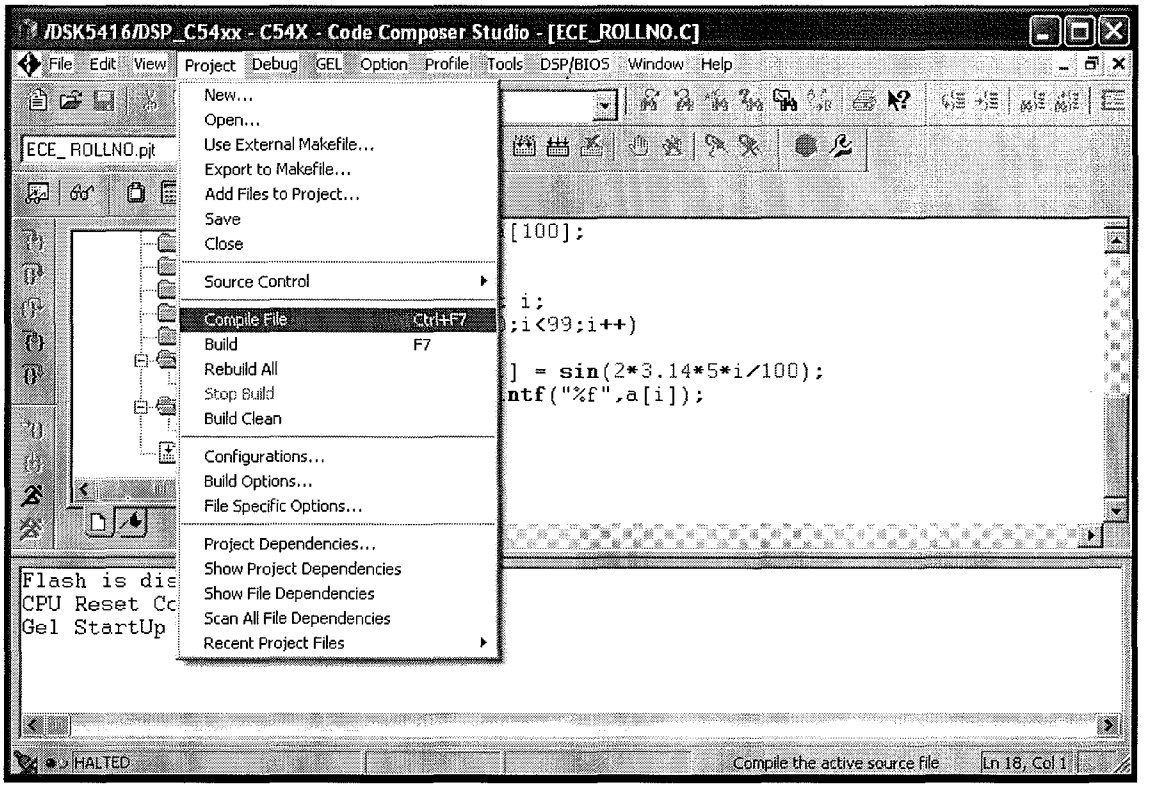
STEP 19: Look in ' CCStudio_v3.1'. Select 'All files' in Files of type and select file 'hello'



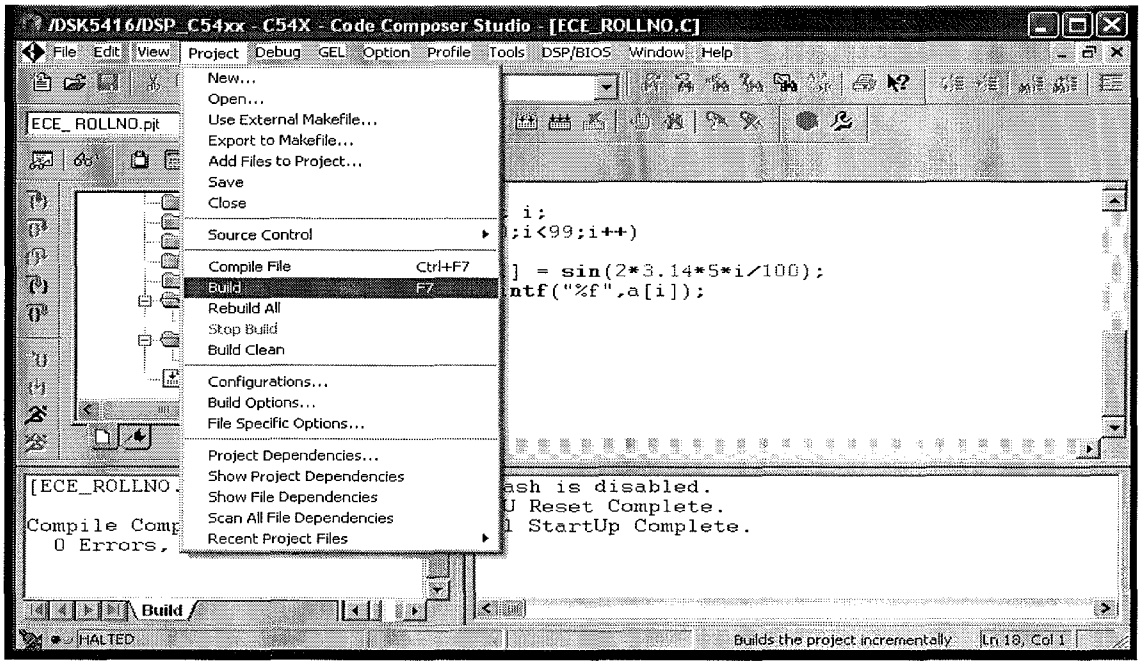
STEP 20: Go to Project – Save



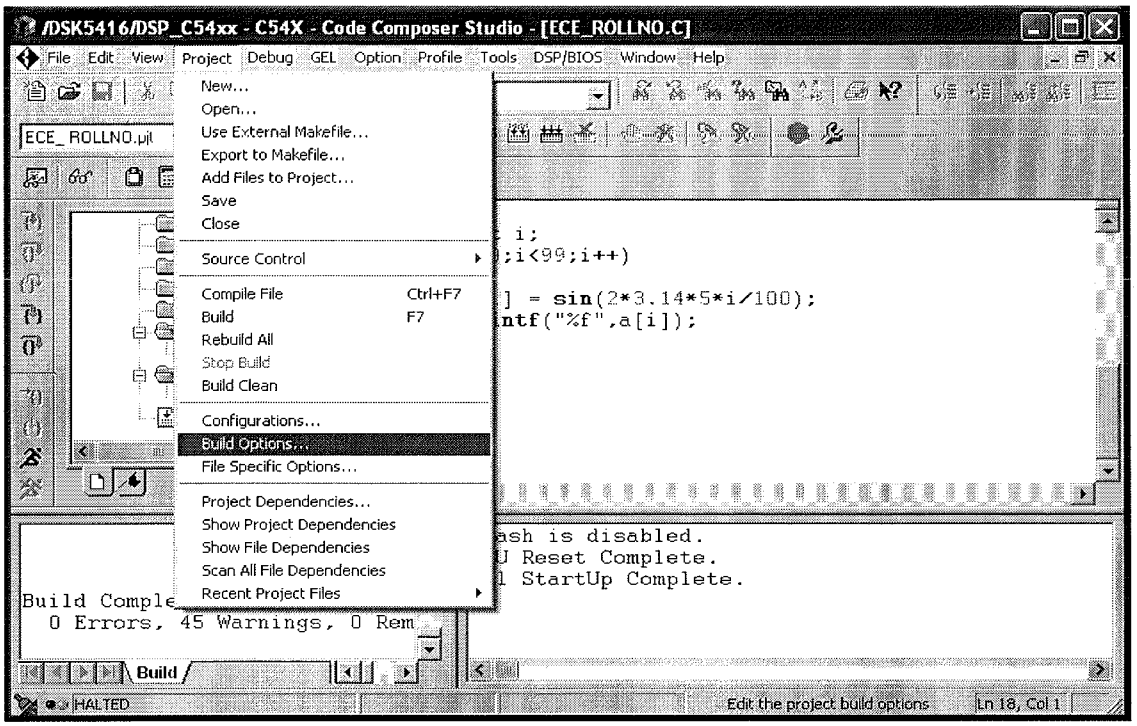
STEP 21: Go to Project – Compile file. Rectify errors if present.



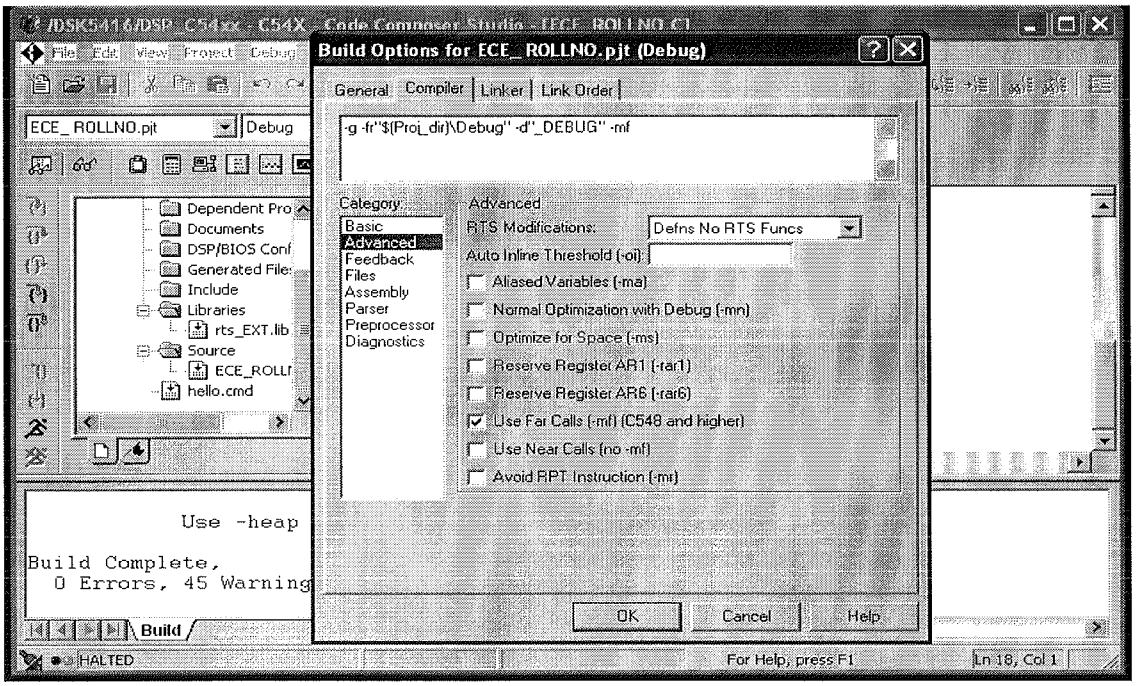
STEP 22: Go to Project – Build.



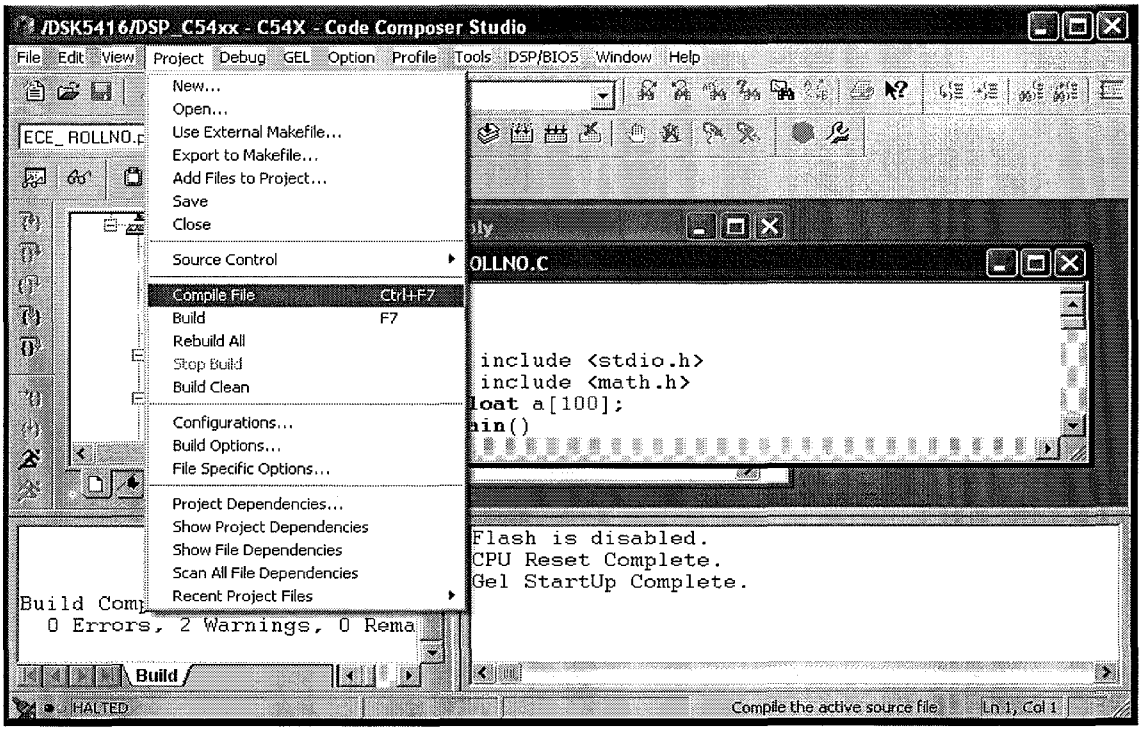
STEP 23: Go to Project – Build options (To reduce the number of warnings)



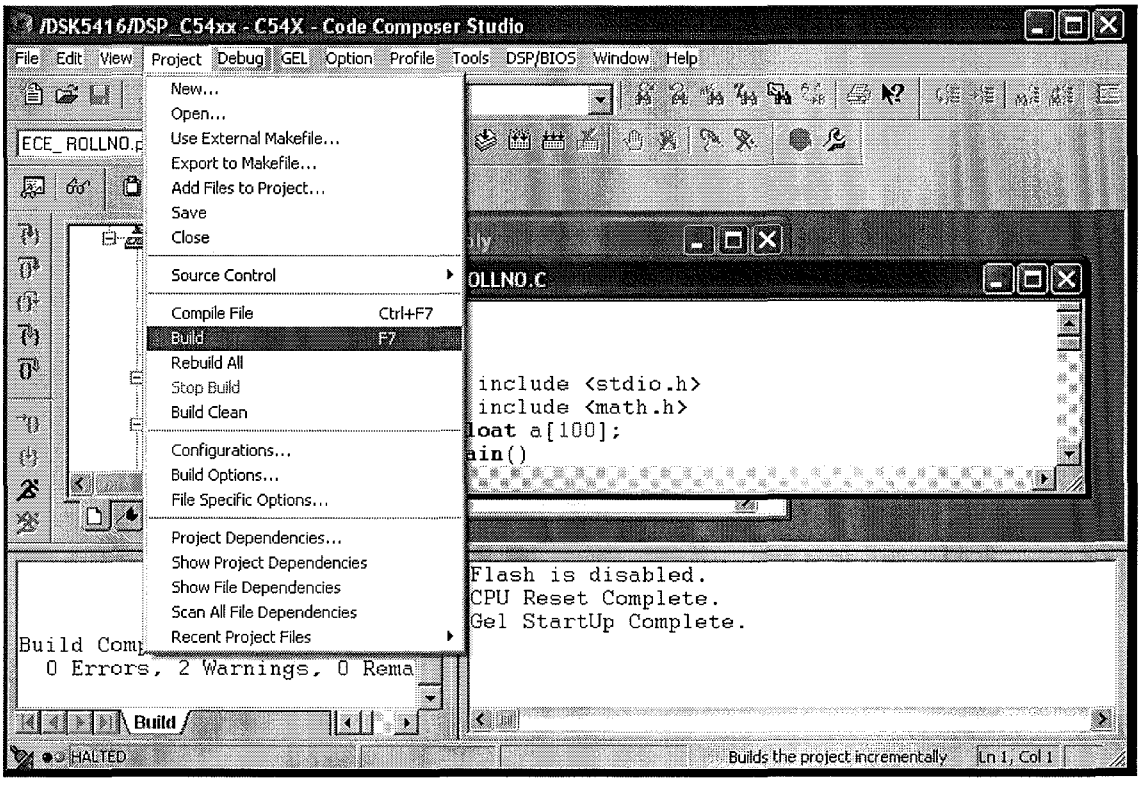
STEP 24: In Build options, select 'Advanced'. Click 'Use Far calls' and then 'OK'.



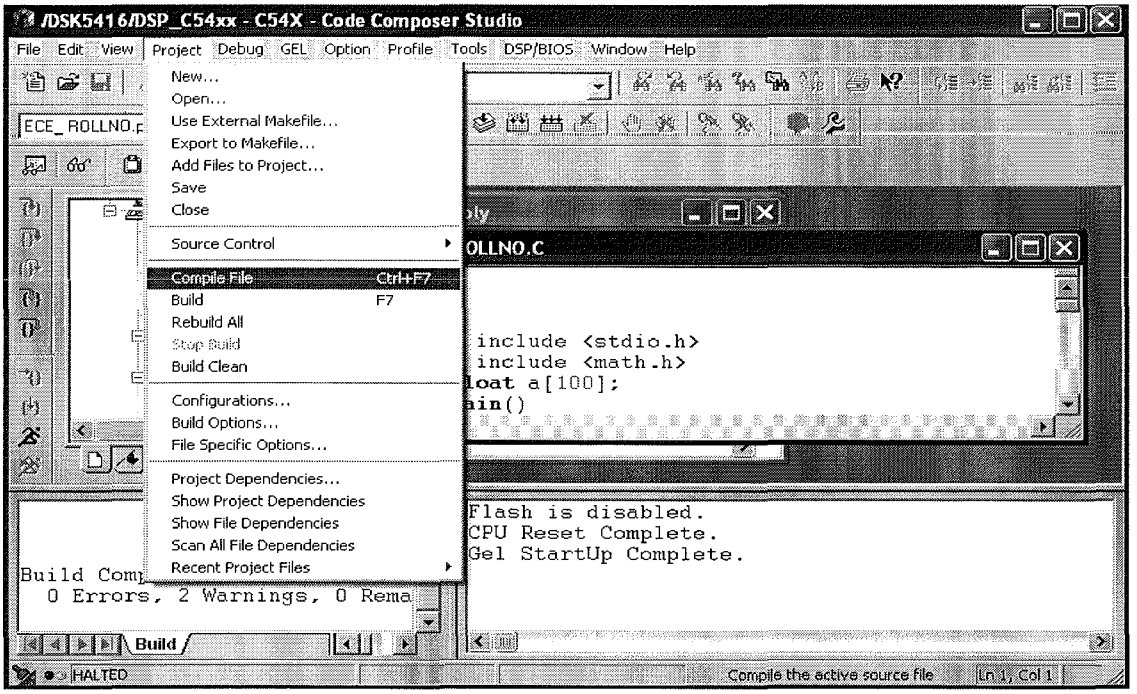
STEP 25: Go to Project – Compile file.



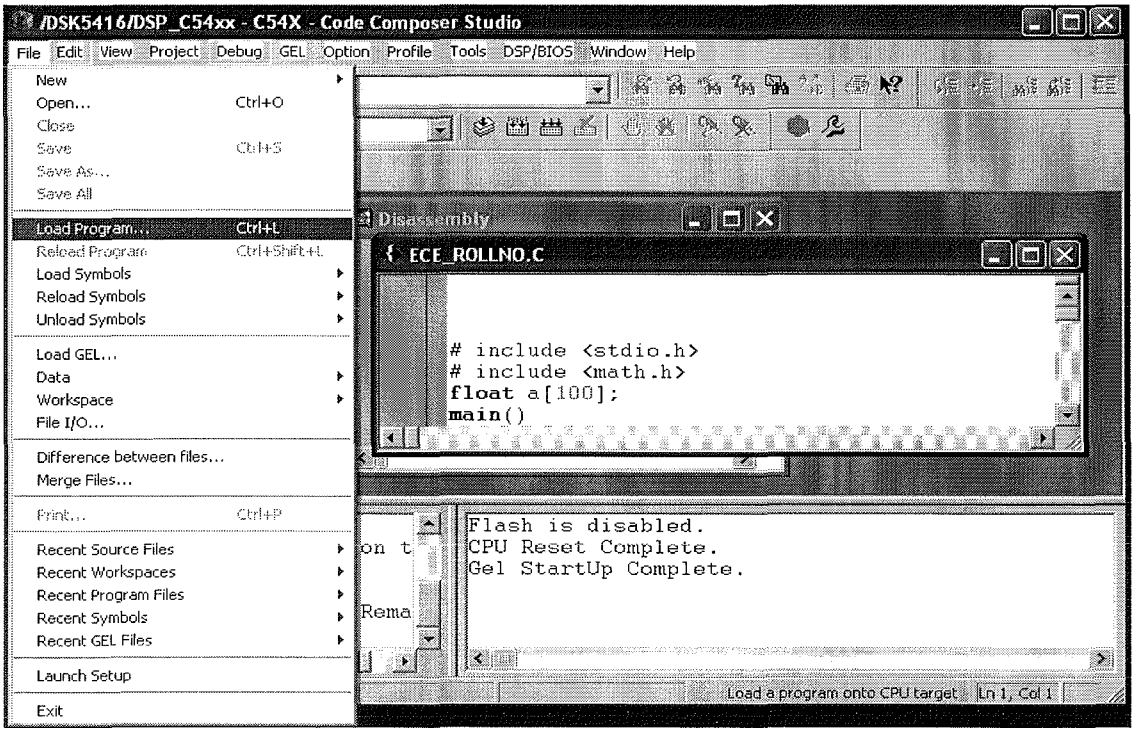
STEP 26: Go to Project – Build.



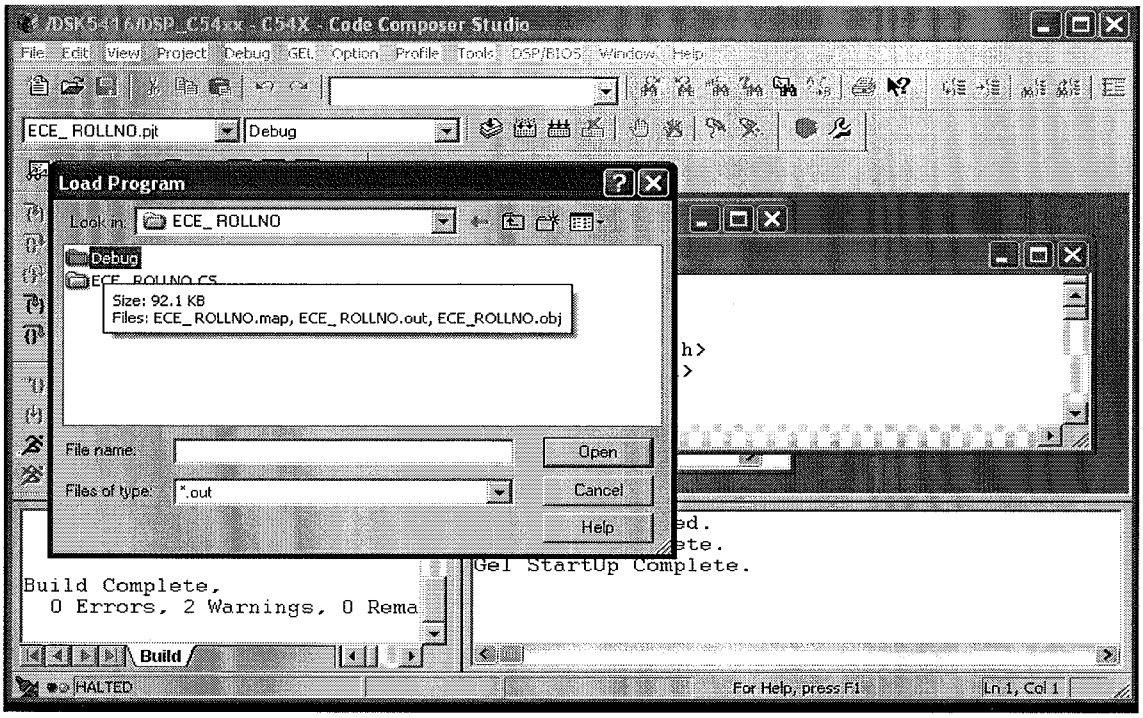
STEP 27: Go to Project – Compile file



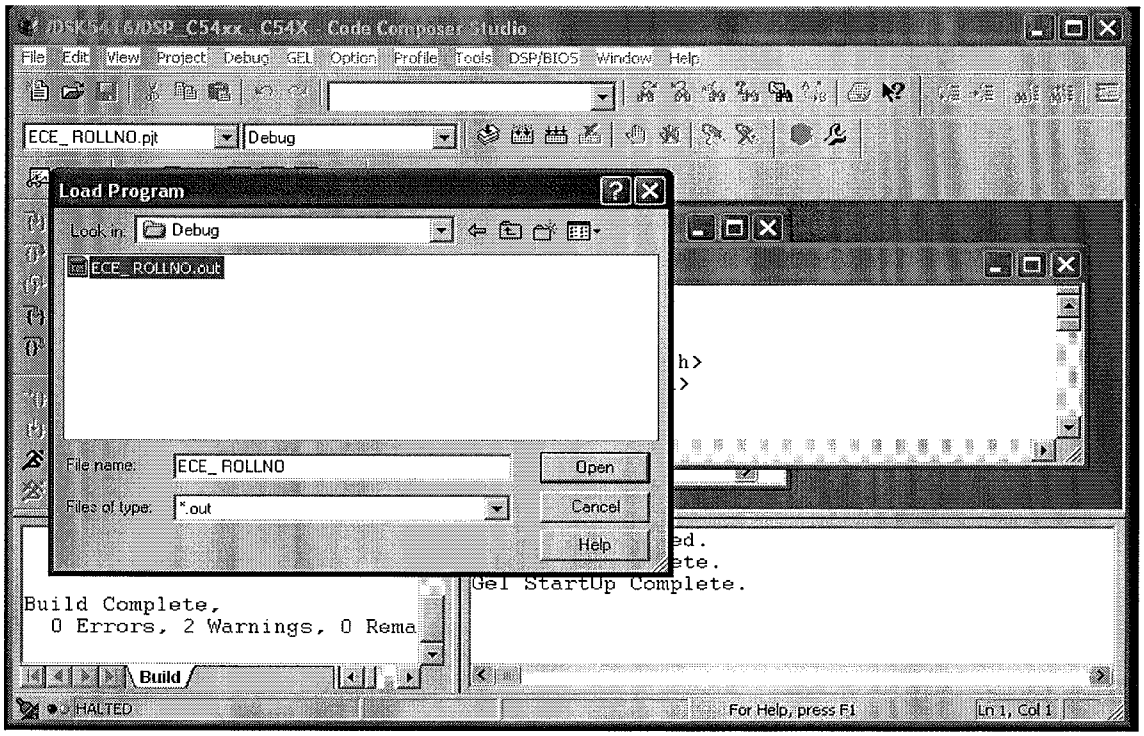
STEP 28: Go to File – Load program.



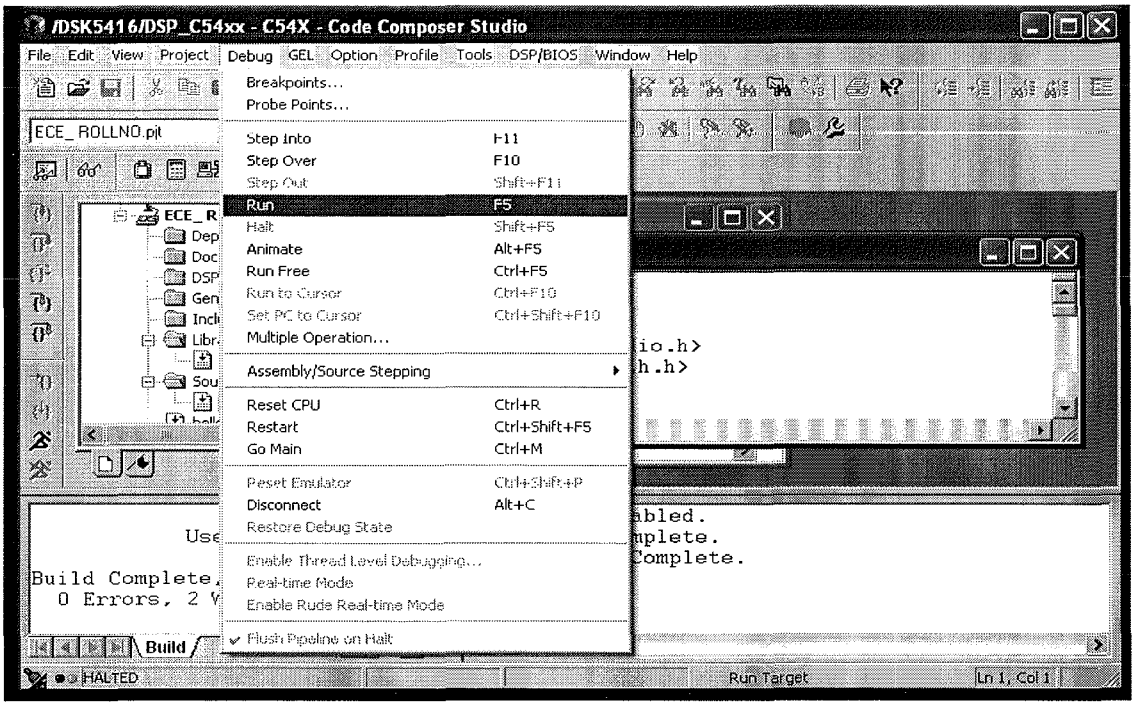
STEP 29: In the Load program window, Look in 'Your Project folder'. Select Debug.



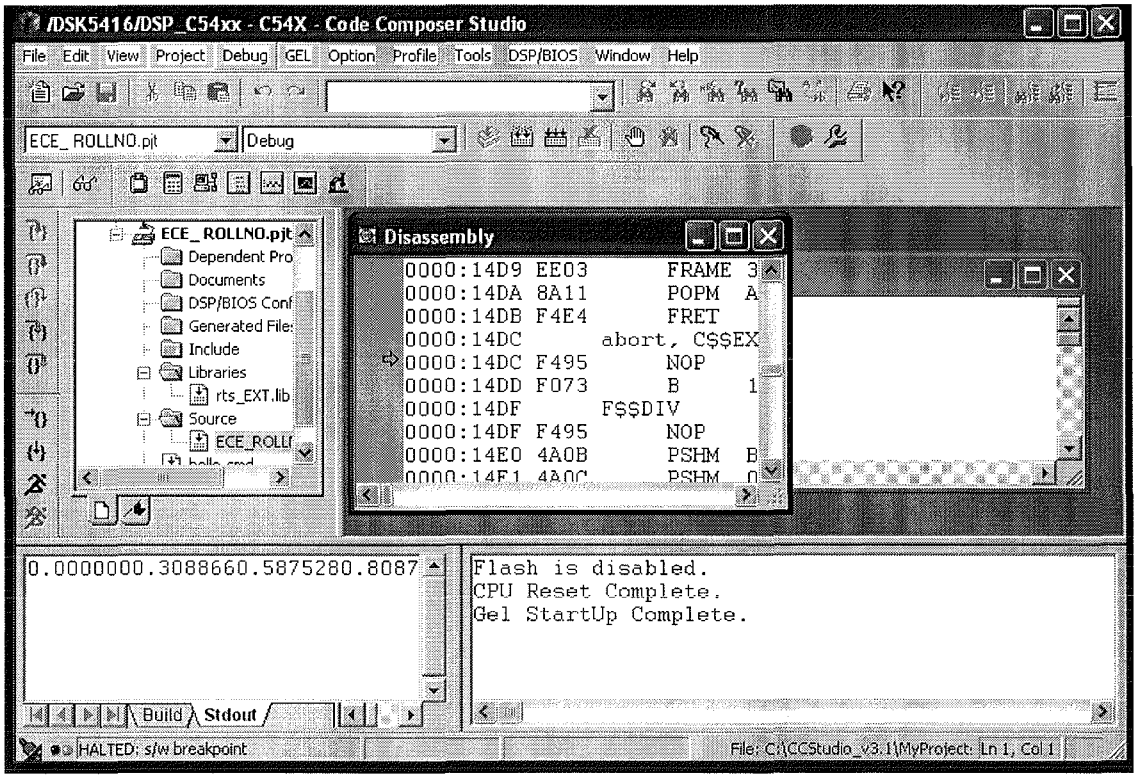
STEP 30: Select the .out file (of your project) and click open.



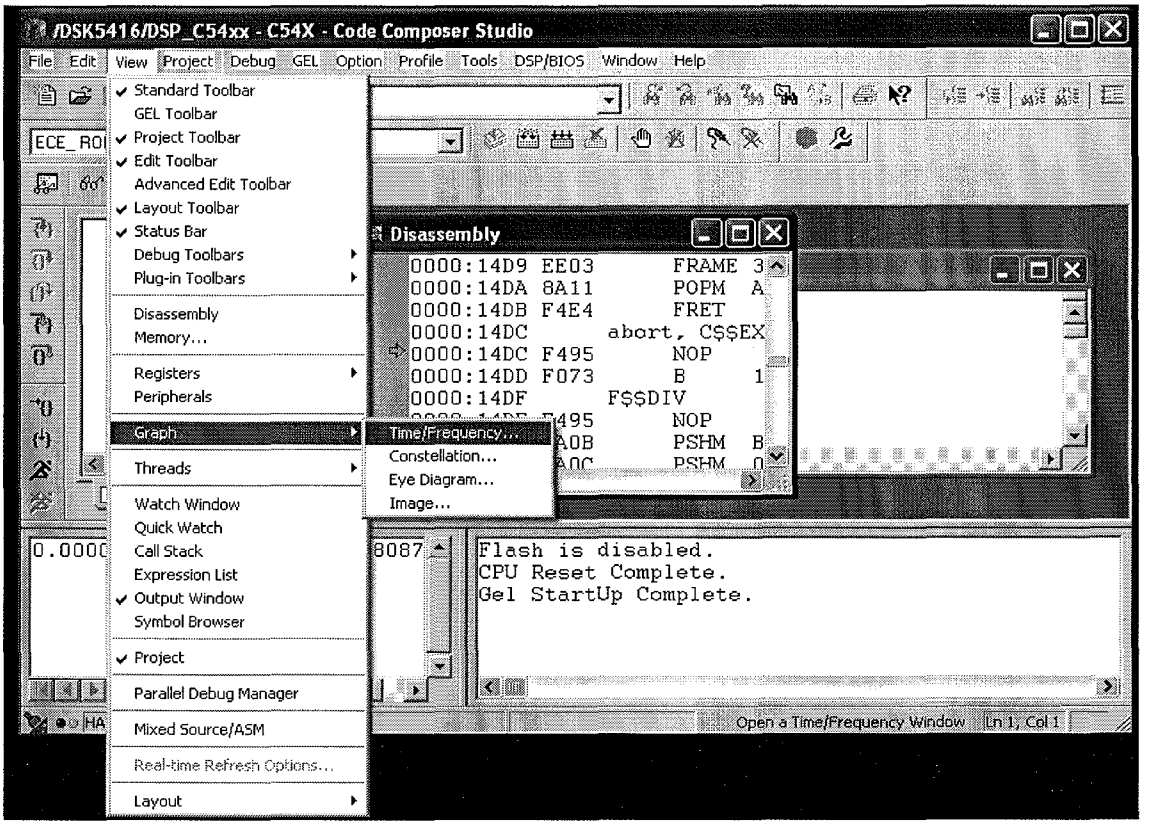
STEP 31: Go to Debug – Run.



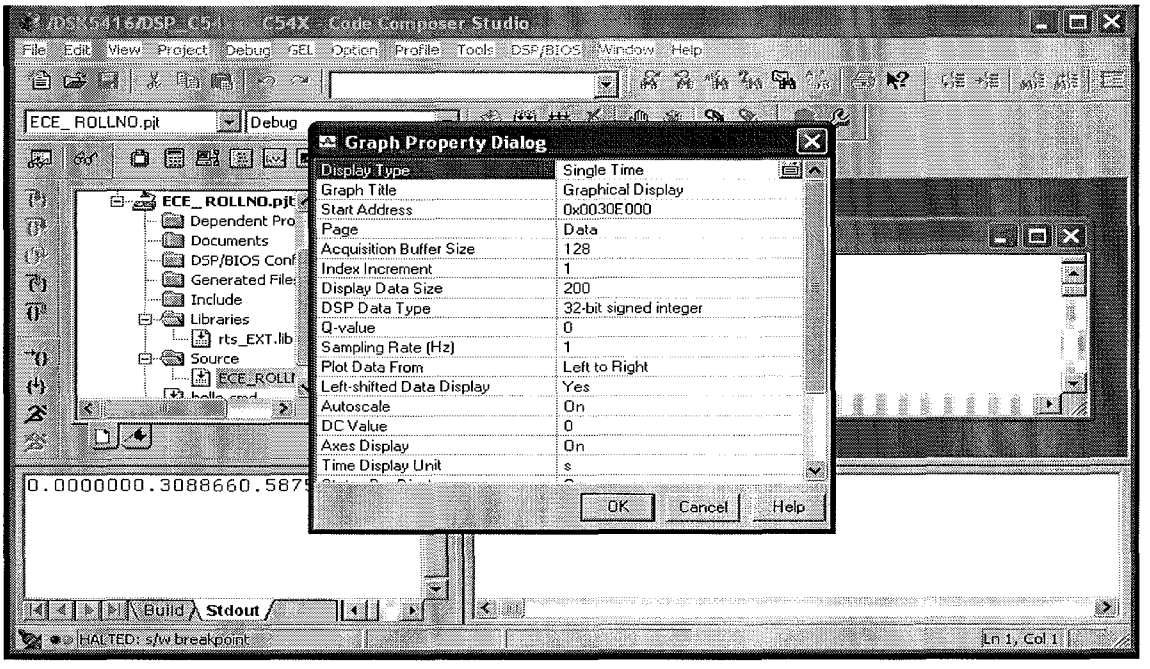
STEP 32: Output is displayed in stdout window.



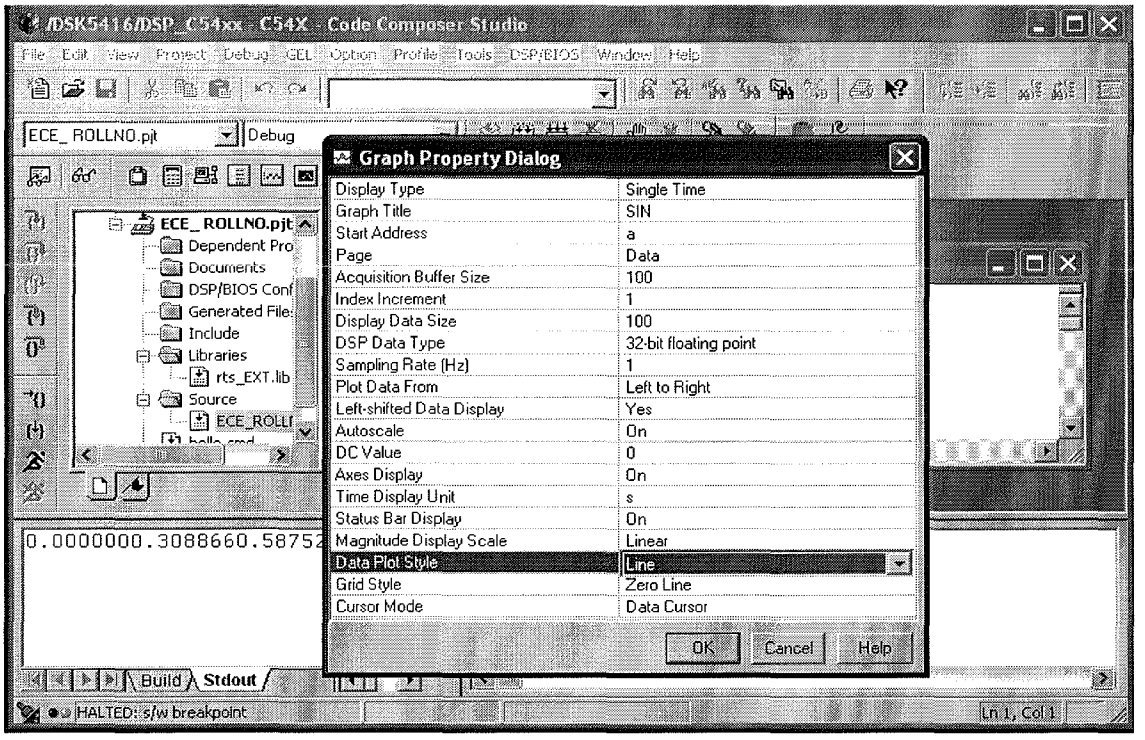
STEP 33: Go to View – Graph – Time / frequency.



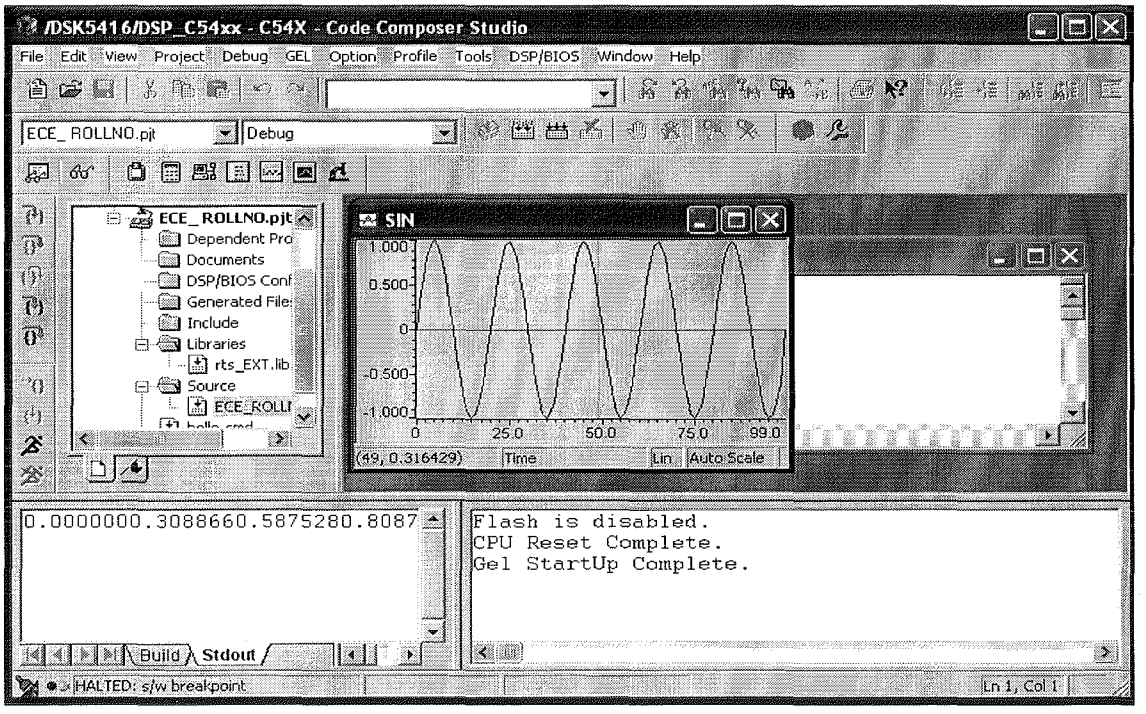
STEP 34: Graph property dialog is displayed.



STEP 35: Change the properties (General properties are given) and click Ok.



STEP 36: Graphical output is displayed.



EC6511
VIVA QUESTIONS

EC 6511 – VIVA QUESTION BANK

1. **What is a Continuous and discrete time signal?**

Continuous time signal:

A signal x(t) is said to be continuous if it is defined for all time t. Continuous time signal arise naturally when a physical waveform such as acoustics wave or light wave is converted into a electrical signal. This is affected by means of transducer. (Example: Microphone, photocell)

Discrete time Signal:

A discrete time signal is defined only at discrete instant of time. The independent variable has discrete values only, which are uniformly spaced. A discrete time signal is often derived from the continuous time signal by sampling it at a uniform rate.

2. **Give the classification of signals?**
   - Continuous- time and discrete time signals
   - Even and ODD signals.
   - Periodic and Non Periodic Signals.
   - Deterministic and Random Signals.
   - Energy and Power Signal.

3. **What are the types of Systems?**
   - Continuous- time and discrete time Systems.
   - Linear and Non-Linear systems.
   - Casual and Non- Causal Systems.
   - Time varying and time in – varying systems.
   - Distributive parameters and Lumped parameters systems.
   - Stable and Un- Stable systems.

4. **What are even and odd signals?**

Even Signal: Continuous time signal x(t) is said to be even if it satisfies the condition.

$X(t)=x(-t)$ for all values of t.

Odd Signal: The signal x(t) is said to be odd if it satisfies the condition.

$X(-t)=x(t)$ for all t.

In other words even signal is symmetric about the time origin or the vertical axis, but odd signals are anti-symmetric about the vertical axis.

5. **What are deterministic and random signals?**

Deterministic signal: Deterministic signal is a signal about which there is no certainty with respect to its value at any time. Accordingly we find that deterministic signals may be modeled as completely specified functions of time.

Random signal: Random signal is a signal about which there is uncertainty before its actual occurrence. Such signal may be viewed as group of signals with each signal in the ensemble having different waveforms. **(e.g)** The noise developed in a television or radio amplifier is an example for random signal.

6. **What are energy and power signal?**

Energy signal: Signal is referred as an energy signal, if and only if the total energy of the signal satisfies the condition 0<E<_ . The total energy of the continuous time signal x (t) is given as E=limT_x2(t)dt, integration limit from –T/2 to +T/2.

Power signal: Signal is said to be powered signal if it satisfies the condition 0<p<_.

The average power of a continuous time signal is given by

P=limt_1/T_x2(t)dt, integration limit is from –T/2 to +T/2.

7. **What are the operations performed on a signal?**

Operations performed on dependent variables:

Amplitude scaling : y(t)=c x(t), where c is the scaling factor, x(t) is the continuous time signal.

Addition : y(t)=x1(t)+x2(t)

Multiplication : y(t)=x1(t)x2(t)

Differentiation : y(t)=d/dt x(t)

Integration : y(t)=_x(t)dt

Operations performed on independent variables.

- Time shifting
- Amplitude scaling
- Time reversal

8. **What are elementary signals and name them?**

The elementary signals serve as a building block for the construction of more complex signals. They are also important in their own right, in that they may be used to model many physical signals that occur in nature. They are five elementary signals. They are as follows.

- Unit step function
- Unit impulse function
- Ramp function
- Exponential function
- Sinusoidal function

9. **What are the properties of a system?**

Stability : A system is said to be stable if the input x(t) satisfies the condition (t) Mx<_ and the output satisfies the condition _y(t)__My<_for all t.

Memory : A system is said to be memory if the output signal depends on the present and the past inputs.

Invertibility: A system is said to be invertible if the input of the system can be recovered from the system output.

Time invariance : A system is said to be time invariant if a time delay or advance of the input signal leads to an identical time shift in the output signal.

Linearity : A system is said to be linear if it satisfies the super position principle

(i.e) R(ax1(t)+bx2(t))=ax1(t)+bx2(t)

10. **What is memory system and memory less system?**

A system is said to be memory system if its output signal at any time depends on the past values of the input signal. Circuit with inductors, capacitors are examples of memory system.

A system is said to be memory less system if the output at any time depends on the present values of the input signal. An electronic circuit with resistors is an example for memory less system.

11. What is an invertible system?

A system is said to be invertible system if the input of the system can be recovered form the system output. The set of operations needed to recover the input as the second system connected in cascade with the given system such that the output signal of the second system is equal to the input signal applied to the system.

$H_{-1}\{y(t)\}=H_{-1}\{H\{x(t)\}\}$

12. What are time invariant systems?

A system is said to be time invariant system if a time delay or advance of the input signal leads to an identical shift in the output signal. This implies that a time invariant system responds identically no matter when the input signal is applied. It also satisfies the condition

$R\{x(n-k)\}=y(n-k)$

13. Is a discrete time signal described by the input output relation $y[n]= r_n x[n]$ time invariant.

A signal is said to be time invariant if $R\{x[n-k]\}= y[n-k]$

$R\{x[n-k]\}=R(x[n]) / x[n]\_x[n-k]= r_n x[n-k]$-------(1)

$Y[n-k]=y[n]/n\_n-k = r_n-kx\ [n-k]$ ----------(2)

Equations (1)_Equation (2).

Hence the signal is time variant.

14. Show that the discrete time system described by the input-output relationship y[n]=nx[n] is linear?

For a sys to be linear $R\{a1x1[n]+b1x2[n]\}=a1y1[n]+b1y2[n]$

L.H.S: $R\{a1x1[n]+b1x2[n]\}=R\{x[n]\}/x[n]\_a1x1[n]+b1x2[n]=a1\ nx1[n]+b1\ nx2[n]$----(1)

R.H.S: $a1y1[n]+b1y2[n]=a1\ nx1[n]+b1\ nx2[n]$----------(2)

Equation (1)= Equation (2)

Hence the system is linear.

15. What is SISO System and MIMO System?

A Control system with single input and single output id referred to as single input single output system. When the number of plant inputs or the number of plant outputs is more than one the system is referred to as multiple input output system. In both the case, the controller may be in the form of a digital computer or microprocessor in which we can speak of the digital control system.

16. What is the output of the system with system function H_1 and H_2 when connected in Cascade and parallel?

When the system with input x(t) is connected in cascade with the system H_1 and H_2 the output of the system is

$$Y(t) = H_2\{H_1\{x(t)\}\}$$

When the system is connected in parallel the output of the system is given by

$$Y(t) = H_1 x_1(t) + \{H_2 x(t).$$

17. What do you mean by periodic and non- Periodic Signals?

A Signal is said to be periodic if X(n +N) = x(n), Where N is the time Signal.
A signal is said to be Non- Periodic if x(n+N)- x(n).

18. Determine the convolution sum of two sequences x(n) = { 3,2,1,2} and h(n) = { 1,2,1,2}.

Y(n) = {3,8,8,12,9,4,4}.

19. Find the convolution of the signals

x(n)	=	1
n	=	-2,0,2
	=	2 n = -1
	=	0 else where .
Y(n)	=	{ 1,1,0,1,-2,0,-1}

20. Determine the solution of the difference equation

$Y(n) = 5/6\ y(n-1)-1/6y(n-2) +x(n)$ for $x(n) =2_n\ u(n)$
$Y(n) = -(1/2)_n u(n) + 2/3(1/3)_n u(n) +8/52_n\ u(n)$

21. Determine the response y(n) , n>=0 of the system described by the second order difference equation.

$y(n) - 4y(n-1) + 4y(n-2) = x(n) -x(n-1)$ when the input is $x(n) = (-1)_n\ u(n)$ and the initial condition are $y(-1) = y(-2)= 1$.

$Y(n) = (7/9 -5/3n)2_n\ u(n) + 2/9(-1)_n\ u(n)$

22. Differentiate DTT and DFT .

DTFT output is continuous in time where as DFT output is Discrete in time.

23. Differentiate between DIT and DIF algorithm.

DIT – Time is decimated and input is bit reversed format output in natural order.
DIF - Frequency is decimated and input is natural order output is bit reversed order.

24. How many stages is there for 8 point D FT?

8

25. How many multiplication terms are required for doing DFT by expressional method and FFT method?

Expression -n2 FFT N/2 log N.

26. Distinguish IIR and FIR filters.

FIR	IIR
Impulse response is finite.	Impulse Response is infinite.
They have perfect linear phase.	They do not have perfect linear phase.
Non recursive	Recursive
Greater flexibility to control the shape of magnitude response	Less response.

27. **Distinguish Analog and Digital filters.**

Analog	Digital
Constructed using active or passive Components and it is described by a differential equation.	Consists of elements like adder, subtractor and delay units and it is described by a difference equation.
Frequency response can be changed by changing the components.	Frequency response can be changed by changing the filter co-efficients.
It processes and Generates analog output	It processes and Generates digital output.
Output varies due to external conditions.	Not influenced by external conditions.

28. **Write the expression for order of Butterworth filter?**

The expression is

N = log (__/ε ½ log(1/k) ½ .

29. **Write the expression for the order of chebyshev filter?**

N = cosh-1(__/e) / cosh-1(1/k).

30. **Write the various frequency transformations in analog domain?**

LPF to LPF: S =' S/ C

LPF to HPF: S = C/S.

LPF to BPF : S = S2xlxu/s (xu-xl)

LPF to BSF: S = s(xu –xl) ? s2 = slxu . X =

31. **Write the steps in designing chebyshev filter?**

- Find the order of the filter.
- Find the value of major and minor axis.
- Calculate the poles.
- Find the denominator function using the above poles.
- The numerator polynomial value depends on the value of n.

  if n is odd: put s=0 in the denominator polynomial.

  If n is even: put s=0 and divide it by (1+e2)1/2

32. **Write down the steps for designing butter worth filter?**

- From the given specifications find the order of the filter.
- Find the transfer function from the value of N.
- Find c
- Find the transfer function ha(s) for the above value of c by su s by that value.

33. **State the equation for finding the poles in chebyshev filter?**

Sk=acos¢k+jbsin¢k, where ¢k=(2+(2k-1)/2n)

34. **State the steps to design digital IIR filter using bilinear method**

Substitute s by 2/T(z-1/z+1), where T=2l_(tan(w/2)) in h(s) to get h(z).

35. **What is warping effect?**

For smaller values of w there exists linear relationship between w and but for larger values of w the relationship is nonlinear. This introduces distortion in the frequency axis. This effect compresses the magnitude and phase response. This effect is called warping effect.

36. **Write a note on pre warping?**

The effect of the non linear compression at high frequencies can be compensated. When the desired magnitude response is piecewise constant over frequency, this compression can be compensated by introducing a suitable rescaling or pre warping the critical frequencies.

37. **Give the bilinear transform equation between s plane and z plane?**

S=2/T (z-1/z+1)

38. **Why impulse invariant method is not preferred in the design of IIR filters other than low pass filter?**

In this method the mapping from s plane to z plane is many to one. Thus there are an infinite number of poles that map to the same location in the z plane, producing an aliasing effect. It is inappropriate in designing high pass filters. Therefore this method is not much preferred.

39. **By impulse invariant method obtain the digital filter transfer function and the differential equation of the analog filter h(s)=1/s+1**

H(z)= 1/1-e-TZ-1
Y/x(s)=1/s+1
Cross multiplying and taking inverse lap lace we get,
D/dt(y(t)+y(t))=x(t)

40. **What is meant by impulse invariant method?**

In this method of digitizing an analog filter, the impulse response of the resulting digital filter is a sampled version of the impulse response of the analog filter. For e.g. if the transfer function is of the form, 1/s-p, then H (z)=1/1-e-pTZ-1

41. **What do you understand by backward difference?**

One of the simplest methods of converting analog to digital filter is to approximate the differential equation by an equivalent difference equation.

d/dt(y(t)/t)=nT=(y(nT)-y(nT-T))/T

42. **What are the properties of chebyshev filter?**

- The magnitude response of the chebyshev filter exhibits ripple either in the stop band or the pass band.
- The poles of this filter lies on the ellipse.

43. **Give the butter worth filter transfer function and its magnitude characteristics for different orders of filter.**

The transfer function of the butter worth filter is given by

H(j_) = 1/1+j(_/_c) N

44. **Give the magnitude function of Butter worth filter.**

The magnitude function of Butter worth filter is

| h(j_)| = 1/[1+(_/_c)2N]1/2, N=1,2,3,4,...

45. **Give the equation for the order N, major, minor axis of an ellipse in case of chebyshev filter?**

The order is given by N=cosh-1(((10.1_p)-1/10.1_s-1)1/2)/cosh-1_s/_p

A=(μ1/N-μ-1/N)/2Ωp

B=Ωp(μ1/N+μ-1/N)/2

46. **Give an expression for poles and zeroes of a chebyshev type2 filters?**

The Zeroes of chebyshev type2 filter SK=j_s/sinK _K, K=1...N

The poles of this filter xK+jyK

xK=_s_K/s2+_K2

yK=_s_K/_s2+_K2_K= a cos_K

47. **How can you design a digital filter from analog filter?**

Digital filter can be designed from analog filter using the following methods.

- Approximation of derivatives
- Impulse invariant method
- Bilinear transformation.

48. **Write down bilinear transformation?**

S=2/T(Z-1/Z+1)

49. **List the butter worth polynomial for various orders.**

N Denominator polynomial

- S+1
- S2+0.707s+1
- (s+1)(s2+s+1)
- (s20.7653s+1)(s2+1.84s+1)
- (s+1)(s2+0.6183s+1)(s2+1.618s+1)
- (s2+1.93s+1)(s2+0.707s+1)(s2+0.5s+1)
- (s+1)(s2+1.809s+1)(s2+1.24s+1)(s2+0.48s+1)

50. **Differentiate Butterworth and Chebyshev filter.**

Butterworth damping factor 1.44 chebyshev 1.06

Butterworth flat response damped response.

51. **What is filter?**

Filter is a frequency selective device which amplify particular range of frequencies and attenuate particular range of frequencies.

52. **What are the types of digital filter according to their impulse response?**

IIR (Infinite Impulse Response) filter

FIR (Finite Impulse Response) filter.

53. **How phase distortion and delay distortion are introduced?**

The phase distortion is introduced when the phase characteristics of a filter is nonlinear within the desired frequency band. The delay distortion is introduced when the delay is not constant within the desired frequency band.

54. **What is mean by FIR filter?**

The filter designed by selecting finite number of samples of impulse response h(n) obtained from inverse fourier transform of desired frequency H(W) are called FIR filters.

55. **Write the steps involved in FIR filter design?**

- Choose the desired frequency response H∂(W)
- Take the inverse fourier transform and obtain H∂(n)
- Convert the infinite duration sequence H∂(n) to h(n).
- Take Z transform of h(n) to get H(Z)

56. **What are the advantages of FIR filter?**

- Linear phase FI filter can be easily designed.
- Efficient realization of FIR filter exists as both recursive and non-recursive structures.
- FIR filter realized non recursively stable.
- The round off noise can be made small in non recursive realization of FIR filter.

57. **What are the disadvantages of FIR filter?**

The duration of impulse response should be large to realize sharp cutoff filters. The non integral delay can lead to problems in some signal processing applications.

58. **What is the necessary and sufficient condition for the linear phase characteristic of a FIR filter?**

The phase function should be a linear function of w, which inturn requires constant group delay and phase delay.

59. **List the well known design technique for linear phase FIR filter design?**

- Fourier series method and window method.
- Frequency sampling method.
- Optimal filter design method.

60. **Define IIR filter?**

The filter designed by considering all the infinite samples of impulse response are called IIR filter.

61. **For what kind of application, the anti symmetrical impulse response can be used?**

The anti symmetrical impulse response can be used to design Hilbert transforms and differentiators.

62. **For what kind of application, the symmetrical impulse response can be used?**

- The impulse response, which is symmetric having odd number of samples can be used to design all types of filters, i.e low pass, high pass, band pass and band reject.
- The symmetric impulse response having even number of samples can be used to design low pass and band pass filter.

63. **What is the reason that FIR filter is always stable?**

FIR filter is always stable because all its poles are at the origin.

64. **What condition on the FIR sequence h (n) are to be imposed n order that this filter can be called a linear phase filter?**

The conditions are

- Symmetric condition h(n)= h(N-1-n)
- Anti symmetric condition h(n)= -h(N-1-n)

65. **Under what conditions a finite duration sequence h(n) will yield constant group delay in its frequency response characteristics and not the phase delay?**

If the impulse response is anti symmetrical, satisfying the condition

H(n)=-h(N-1-n)

The frequency response of FIR filter will have constant group delay and the phase delay.

66. **State the conditions for a digital filter to be causal and stable?**

- A digital filter is causal if its impulse response h(n)=0 for n<0.
- A digital filter is stable if its impulse response is absolutely summable, i.e ∞—n=-∞ | h(n) | < ∞

67. **What are the properties of FIR filter?**

- FIR filter is always stable.
- A realizable filter can always be obtained.
- FIR filter has a linear phase response.

68. **When cascade from realization is preferred in FIR filters?**

The cascade from realization is preferred when complex zeros with absolute magnitude less than one.

69. **What are the disadvantages of Fourier series method?**

In designing FIR filter using Fourier series method the infinite duration impulse is truncated at n=+_(N-1/2).Direct truncation of the series will lead to fixed percentage overshoots and undershoots before and after an approximated discontinuity in the frequency response.

70. **What is Gibbs phenomenon? Or what are Gibbs oscillations?**

One possible way of finding an FIR filter that approximates H(ejώ) would be to truncate the infinite Fourier series at n=+_(N-1/2). Abrupt truncation of the series will lead to oscillation both in pass band and is stop band. This phenomenon is known as Gibbs phenomenon.

71. **What are the desirable characteristics of the windows?**

The desirable characteristics of the window are

- The central lobe of the frequency response of the window should contain most of the energy and should be narrow.
- The highest side lobe level of the frequency response should be small.
- The sides of the frequency response should decrease in energy rapidly as ώ tends to Π

72. **Compare Hamming window with Kaiser Window.**

Hamming window with Kaiser Window

- The main lobe width is equal to 8Π/N and the peak side lobe level is -41dB.
- The low pass FIR filter designed will have first side lobe peak of -53 dB
- The main lobe width, the peak side lobe level can be varied by varying the parameter α and N
- The side lobe peak can be varied by varying the parameter α.

73. **What is the necessary and sufficient condition for linear phase characteristics in FIR filter?**

The necessary and sufficient condition for linear phase characteristics in FIR filter is the impulse response h(n) of the system should have the symmetry property, i.e,

H(n)=h(N-1-n)

Where N is the duration of the sequence

74. **What are the advantages of Kaiser Window?**

- It provides flexibility for the designer to select the side lobe level and N.
- It has the attractive property that the side lobe level can be varied continuously from the low value in the Blackman window to the high value in the rectangle window.

75. **What is the principle of designing FIR filter using frequency sampling method?**

In frequency sampling method the desired magnitude response is sampled and a linear phase response is specified. The samples of desired frequency response are defined as DFT coefficients. The filter coefficients are then determined as the IDFT of this set of samples.

76. **For what type of filters frequency sampling method is suitable?**

Frequency sampling method is attractive for narrow band frequency selective filters where only a few of the samples of the frequency response are non-zero.

77. **What is meant by autocorrelation?**

The autocorrelation of a sequence is the correlation of a sequence with it shifted version, and this indicates how fast the signal changes.

78. **Define white noise?**

A stationary random process is said to be white noise if its power density spectrum is constant .Hence the white noise has flat frequency response spectrum.

Sx(w)=_x2,-__w_

79. **What do you understand by a fixed-point number?**

In fixed point arithmetic the position of the binary point is fixed. The bit to the right represents the fractional part of the number and those to the left represent the integer part. For example, the binary number 01.1100 has the value 1.75 in decimal.

80. **What is the objective of spectrum estimation?**

The main objective of spectrum estimation is the determination of the power spectral density of a random process. The estimated PSD provides information about the structure of the random process which can be used for modeling, prediction or filtering of the deserved process.

81. **List out the addressing modes supported by C5X processor?**
- Direct addressing
- Indirect addressing
- Immediate addressing
- Dedicated- register addressing
- Memory- mapped register addressing
- Circular addressing

82. **What is meant by block floating point representation? What are its advantages?**

In block point arithmetic the set of signals to be handled is divided into blocks. Each block has the same value for the exponent. The arithmetic operations with in the block uses fixed point arithmetic and only one exponent per block is stored thus saving memory. This representation of numbers is more suitable in certain FFT flow graph and in digital audio applications.

83. **What are the advantages of floating point arithmetic?**
- Large dynamic range.
- Overflow in floating point representation is unlike.

84. **What are the three quantization errors to finite word length registers in digital filters?**
- Input quantization error
- Coefficient quantization error
- Product quantization error

85. **How the multiplication and addition are carried out in floating point arithmetic?**

In floating point arithmetic, multiplication is carried out as follows. Let f1=M1*2c1 and f2=M2*2c2. Then f3=f1*f2=(M1*M2) 2(c1+c2). That is, mantissa is multiplied using fixed-point arithmetic and the exponents are added. The sum of two floating-point numbers is carried out by shifting the bits of the mantissa of the smaller number to the right until the exponents of the two numbers are equal and then adding the mantissas.

86. **What do you understand by input quantization error?**

In digital signal processing, the continuous time input signals are converted into digital using a b-bit ACD. The representation of continuous signal amplitude by a fixed digit produce an error, which is known as input quantization error.

87. **List the on-chip peripherals in 5X.**

The C5X DSP on-chip peripherals available are as follows.
- Clock Generator
- Hardware Timer
- Software-Programmable Wait-State Generators
- Parallel I/O Ports
- Host Port Interface(HPI)
- Serial Port
- Buffered Serial Port(BSP)
- Time –Division Multiplexed (TDM) Serial Port
- User Maskable Interrupts

88. **What is the relationship between truncation error e and the bits b for representing a decimal into binary?**

For a 2's complement representation, the error due to truncation for both positive and negative values of x is 0>=xt-x>-2-b where b is the number of bits and xt is the truncated value of x. The equation holds good for both sign magnitude, 1's complement if x>0. If x<0, then for sign magnitude and for 1's complement the truncation error satisfies

89. **What is meant by rounding? Discuss its effect on all types of number representation?**

Rounding a number to b bits is accomplished by choosing the rounded result as the b bit number closest to the original number un rounded. Or fixed point arithmetic, the error made by rounding a number to b bits satisfy the inequality -2-b 2-b----<=xt<=-----2 2 for all three types of number systems, i.e., 2's complement, 1's complement and sign magnitude. For floating point number the error made by rounding a number to b bits satisfy the inequality -2-b<=E<=2-b where E=xt-x-----x

90. **What is meant by A/D conversion noise?**

A DSP contains a device, A/D converter that operates on the analog input x(t) to produce xq (t) which is binary sequence of 0s and 1s. At first the signal x(t) is sampled at regular intervals to produce a sequence x(n) is of infinite precision. Each sample x(n) is expressed in terms of a finite number of bits given the sequence xq(n). The difference signal e(n)=xq (n)-x(n) is called A/D conversion noise.

91. **What is the effect of quantization on pole location?**

Quantization of coefficients in digital filters lead to slight changes in their value. This change in value of filter coefficient modify the pole zero locations. Sometimes the pole locations will be changed in such a way that the system may drive into instability.

92. **Which realization is less sensitive to the process of quantization?**

Cascade form.

93. **What is meant by quantization step size?**

Let us assume a sinusoidal signal varying between +1 and -1 having a dynamic range 2. If the ADC used to convert the sinusoidal signal employs b+1 bits including sign bit, the number of levels sq=2 = 2-b/2b+1 where q is known as quantization step size.

94. **How would you relate the steady-state noise power due to quantization and the b bits representing the binary sequence?**

Steady state noise power where b is the number of bits excluding sign bit.

95. **What is overflow oscillation?**

The addition of two fixed –point arithmetic numbers because overflow the sum exceeds the word size available to store the sum. This overflow caused by adder make the filter output to oscillate between maximum amplitude limits. Such limit cycles have been referred to as over flow oscillations.

96. **What are the methods used to prevent overflow?**

There are two methods used to prevent overflow.

- Saturation arithmetic
- Scaling

97. **What are the two kinds of limit cycle behavior in DSP?**

- Zero input limit cycle oscillations
- Overflow limit cycle oscillations

98. **What is meant by "dead band" of the filter?**

The limit cycle occur as a result of quantization effect in multiplication. The amplitudes of the output during a limit cycle are confined to a range of values called the dead band of the filter.

99. **Explain briefly the need for scaling in the digital filter implementation.**

To prevent overflow, the signal level at certain points in the digital filter must be scaled so that no overflow occurs in the adder.

100. **What are the different buses of TMS320C5X and their functions?**

The C5X architecture has four buses and their functions are as follows:

Program bus (PB): It carries the instruction code and immediate operands from program memory space to the CPU.

Program address bus (PAB): It provides addresses to program memory space for both reads and writes.

Data read bus (DB): It interconnects various elements of the CPU to data memory space.

Data read address bus (DAB): It provides the address to access the data memory space.

CONTENT BEYOND

SYLLABUS

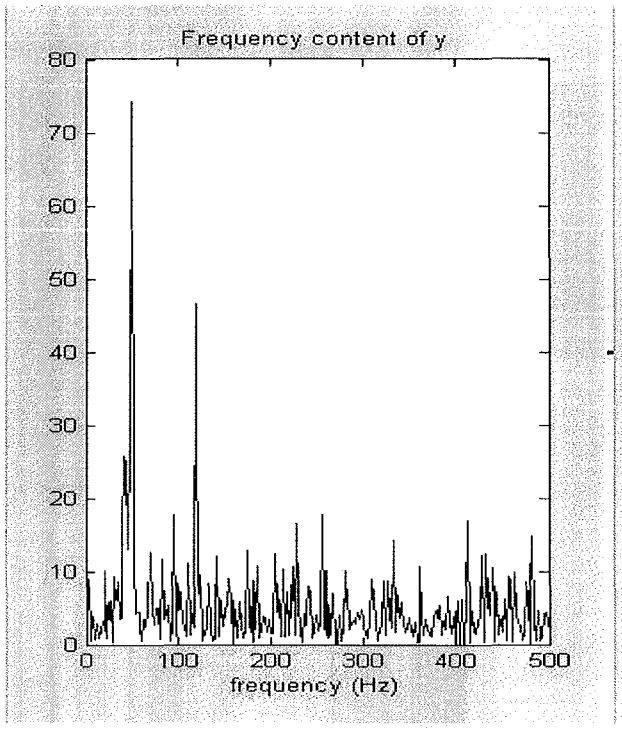
COMPUTATION OF POWER DENSITY SPECTRUM OF A SEQUENCE

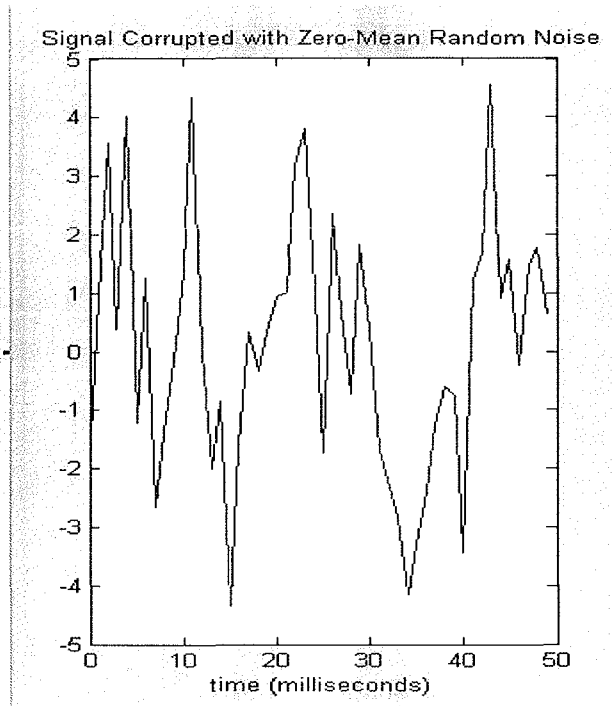
AIM: To compute Power Density Spectrum of a sequence using MATLAB.

PROGRAM:

```
t = 0:0.001:0.6;
x = sin(2*pi*50*t)+sin(2*pi*120*t);
y = x + 2*randn(size(t));
figure, plot(1000*t(1:50),y(1:50)) ;
title('Signal Corrupted with Zero-Mean Random Noise');
xlabel('time (milliseconds)');
Y = fft(y,512);
%The power spectral density, a measurement of the energy at various frequencies, is:
Pyy = Y.* conj(Y) / 512;
f = 1000*(0:256)/512;
figure, plot(f,Pyy(1:257));
title('Frequency content of y');
xlabel('frequency (Hz)');
```

OUTPUT:





RESULT:

Hence the Power Density Spectrum of a given sequence using MATLAB is computed successfully.

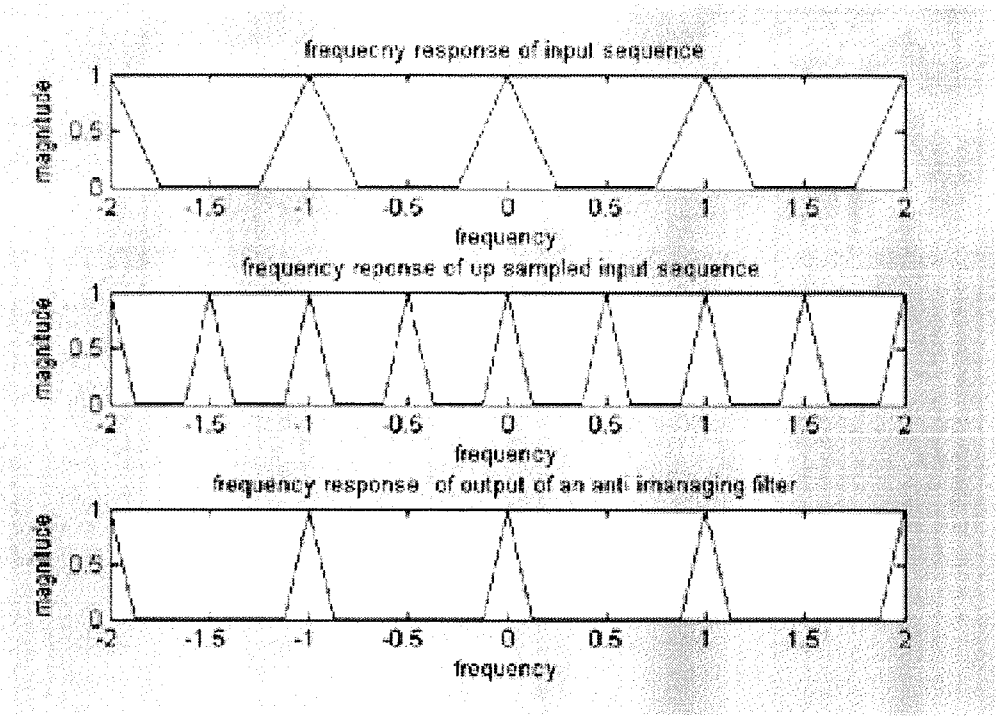
FREQUENCY RESPONSE OF ANTI IMAGING AND ANTI ALIASING FILTERS

AIM: To observe the frequency responses of anti imaging and anti aliasing filters.

PROGRAM FOR ANTI IMAGING FILTER

```
clear all;
n=0:1:1023;
x=1/4*sinc((1/4)*(n-512)).^2
i=1:1024;
y=[zeros(1,2048)];
y(2*i)=x;
f=-2:1/512:2;
h1=freqz(x,1,2*pi*f);
h2=freqz(y,1,2*pi*f);
subplot(3,1,1);
plot(f,abs(h1));
xlabel('frequency');
ylabel('magnitude');
title('frequecny response of input sequence');
subplot(3,1,2);
plot(f,abs(h2));
xlabel('frequency');
ylabel('magnitude');
title('frequency reponse of up sampled input sequence ');
p=fir1(127,.3);
xf=filter(p,1,y);
h4=freqz(xf,1,2*pi*f);
subplot(3,1,3);
plot(f,abs(h4));
title('frequency response of output of an anti imanaging filter');
xlabel('frequency');
ylabel('magnitude');
```

OUTPUT:



PROGRAM FOR ANTI ALIASING FILTER:

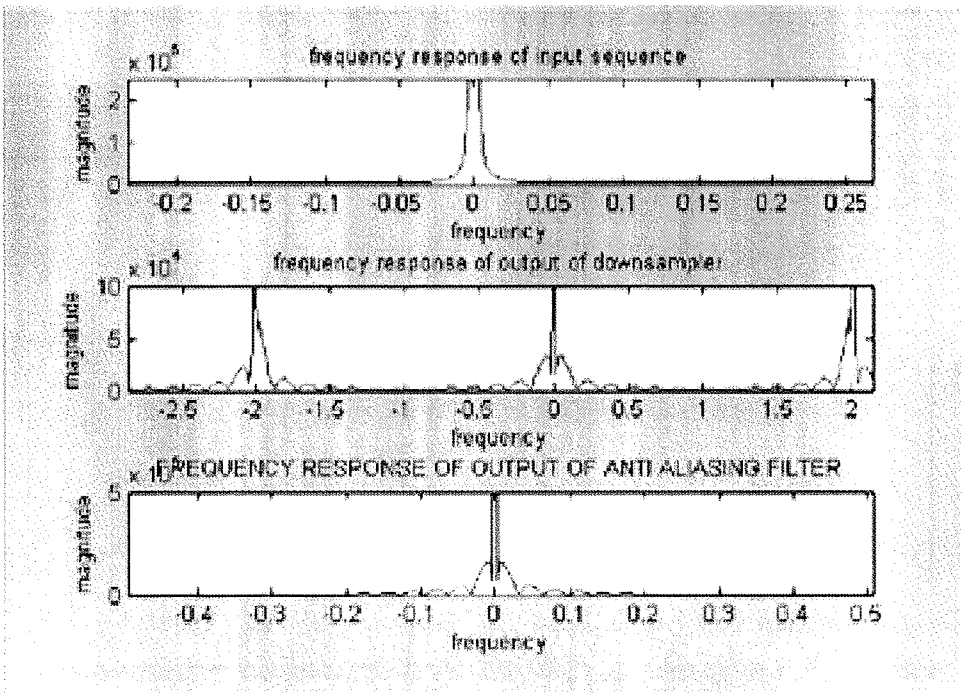
```
clear all;
F=input('enter the highest normalized frequency
component'); D=input('emter the dicimation
factor'); n=0:1:1024;
xd=(F/2*sinc(F/2)*(n-512)).^2;
f=-2:1/512:2;
h1=freqz(xd,1,pi*f);
subplot(3,1,1);
plot(f,abs(h1));
xlabel('frequency');
ylabel('magnitude');
title('frequency response of input sequence');
if(F*D<=1)
   xd1=F/2*sinc(F/2*(n-512)*D).^2;
   h2=freqz(xd1,1,pi*f*D);
   subplot(3,1,3)
  plot(f,abs(h2));
  axis([-2 2 0 1]);
  h2=freqz(xd1,1,pi*f*D);
  subplot(3,1,2);
  plot(f*D,abs(h2));
  axis([-2*D 2*D 0 1]);
else
```

```
  p=fir1(127,1/D);
  xf=filter(p,1,xd);
  h4=freqz(xf,1,pi*f);
  subplot(3,1,3);
  plot(f,abs(h4));
  title('FREQUENCY RESPONSE OF OUTPUT OF ANTI
  ALIASING FILTER'); xlabel('frequency');
ylabel('magnitude');
i=1:1:1024/D;
xr=xf(i*D);
h5=freqz(xr,1,pi*f*D);
subplot(3,1,2);
plot(f*D,abs(h5));
title('frequency response of output of downsampler');
xlabel('frequency');
ylabel('magnitude');
end;
```

EXAMPLE:

Enter the highest normalized frequency component 0.25
Enter the decimation factor 5

OUTPUT:



RESULT:

Hence the frequency responses of anti imaging and anti aliasing filters are observed successfully.

CD DATA TO DVD DATA

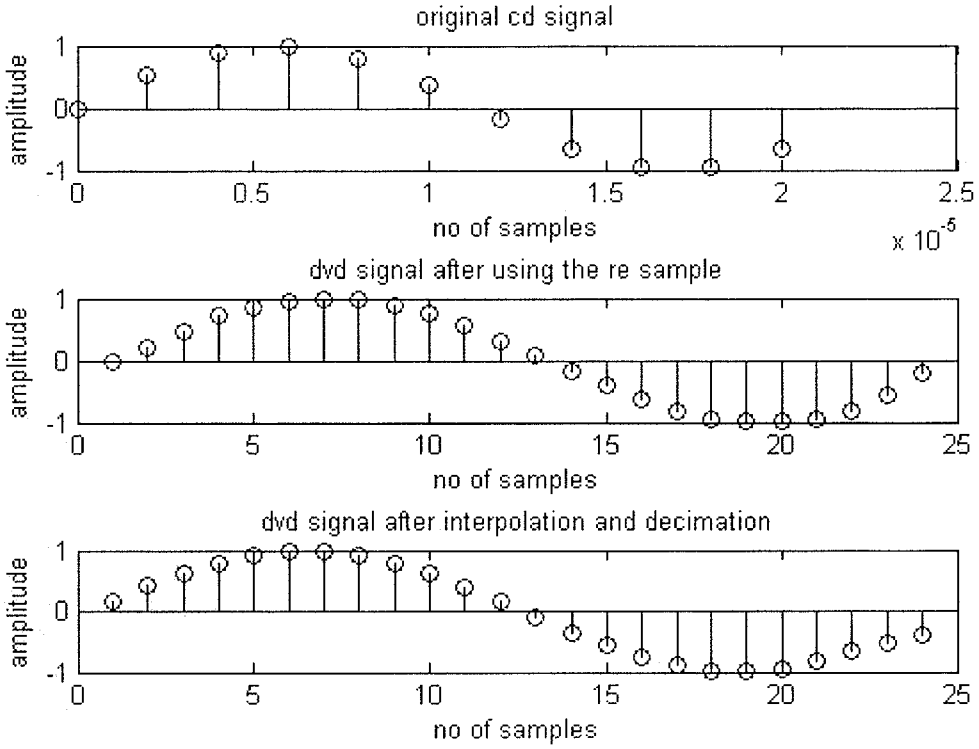
AIM:

To convert CD data to DVD data.

PROGRAM

```
clear all;
fc=44100;
t=0:0.000002:0.00002;
x=sin(2*pi*fc*t);
subplot(3,1,1);
stem(t,x);
title('original cd signal');
xlabel('no of samples');
ylabel('amplitude');
i=13;
d=6;
y=resample(x,i,d);
subplot(3,1,2);
stem(y);
title('dvd signal after using the re sample');
xlabel('no of samples');
ylabel('amplitude');
in=interp(x,i);
de=decimate(in,d);
subplot(3,1,3);
stem(de);
title('dvd signal after interpolation and decimation');
xlabel('no of samples');
ylabel('amplitude');
```

OUTPUT:



RESULT:

Hence the CD data is converted into DVD data successfully.

STUDY OF PROPERTY OF LINEAR TIME INVARIANT SYSTEM

AIM:

To study the stability property of a LTI system.

PROGRAM

```
clc;
clear all;
close all;
b=input('enter the denominator coefficient of the filter');
k=poly(b);
knew=fliplr(k);
s=all(abs(knew));
if(s==1)
disp("'stable system'");
else
disp("'Non stable system'");
end
```

OUTPUT

```
enter the denominator coefficient of the filter [1 -1 0.5]
'stable system'
```

RESULT

Hence the stability property of a LTI system is studied successfully.

EXERCISE

Exercise :1

1. Write a MATLAB program to generate of Sine sequence in Discrete form.
2. Write a MATLAB program to generate of Cosine sequence in Discrete form.
3. Write a MATLAB program to perform addition of two Sinusoidal sequence in Discrete form.
4. Write a MATLAB program to perform addition of two Cosinusoidal sequence in Discrete form.
5. Write a MATLAB program to generate of Sine sequence with period 5 in Discrete form.
6. Write a MATLAB program to generate of Sine sequence with period 10 in Discrete form.
7. Write a MATLAB program to perform growing exponential signal in complex form.
8. Write a MATLAB program to perform decaying exponential signal in complex form.
9. Write a MATLAB program to generate decaying exponential signal.
10. Write a MATLAB program to perform basic mathematical operation called addition on signals.
11. Write a MATLAB program to perform basic mathematical operation called subtraction on signals.
12. Write a MATLAB program to perform basic mathematical operation named multiplication on signals.
13. Write a MATLAB program to perform scaling (amplitude and time scaling) on Discrete time signals.
14. Write a MATLAB program to perform folding on Discrete time signals.
15. Write a MATLAB program to perform shifting of Discrete time signals by Right shift (or advance).
16. Write a MATLAB program to perform shifting of Discrete time signals by left shift (or delay).

Exercise :2

1. Write a MATLAB program to perform circular convolution of the given Discrete time sequence $x_1(n)$ and $x_2(n)$.

   $x_1(n)=1$ for $1<n<10$

   $x_2(n)=1$ for $2<n<10$

2. Write a MATLAB program to perform circular convolution of the discrete time sequence

   $x_1(n)=\{0,1,0,1\}$

   $x_2(n)=\{1,2,1,2)$

Exercise :3

1. Write a MATLAB program to perform 8 point DFT of the discrete time sequence x(n)={2,1,2,1,1,2,1,2} and sketch the magnitude and phase spectrum.

2. Write a MATLAB program to perform inverse DFT. Use the output of exercise 1 as input.

3. Write a MATLAB program to perform 4 point DFT of the discrete time sequence x(n)={1,1,2,3} using FFT and sketch the magnitude and phase spectrum.

Exercise :4

1. Write a MATLAB program to design a Butterworth Analog Low Pass Filter.
2. Write a MATLAB program to design a Butterworth Digital/ Analog High Pass Filter.
3. Write a MATLAB program to design a Butterworth Digital/ Analog Band pass filter.
4. Write a MATLAB program to design a Butterworth Digital/ Analog Band Stop Filter.
5. Write a MATLAB program to design a Chebyshev Analog/ Digital Low Pass Filter.
6. Write a MATLAB program to design a Chebyshev Analog/ Digital High Pass Filter.
7. Write a MATLAB program to design a Chebyshev Analog/ Digital Band Pass Filter.
8. Write a MATLAB program to design a Chebyshev Analog/ Digital Band Stop Filter.
9. Write a MATLAB program to convert analog filter in to digital filter using Impulse Invariant transformation. Consider numerator coefficient and denominator coefficient of analog filter as [1,2] and [1,5,11,15] respectively.
10. Write a MATLAB program to convert analog filter in to digital filter using Bilinear transformation. Consider numerator coefficient and denominator coefficient of analog filter as [2] and [1,3,2] respectively.

Exercise :5

1. Write a MATLAB program to downsample the signal x(n) by sampling rate reduction factor i) 3 ii) 4.
2. Write a MATLAB program to upsample the signal x(n) by sampling rate multiplication factor i) 3 ii) 4.
3. Write a MATLAB program to illustrate the effect of downsampling in frequency domain of a signal for sampling rate reduction factor
   i) 2 ii) 3 iii) 4.
4. Write a MATLAB program to illustrate the effect of upsampling in frequency domain of a signal for sampling rate multiplication factor
   i) 2 ii) 3 iii) 4.