**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**
**VIỆN ĐIỆN TỬ - VIỄN THÔNG**
**BỘ MÔN MẠCH VÀ XỬ LÝ TÍN HIỆU**

---


# BÁO CÁO THÍ NGHIỆM
# XỬ LÝ SỐ TÍN HIỆU

**Sinh viên thực hiện: Nguyễn Minh Hiếu**

**Lớp: Điện tử 03 – K60**

**Mssv: 20151336**

**Mã lớp thí nghiệm: 678825**

**Hà Nội – 2018**

# BÀI 1. Mô phỏng hệ thống và tín hiệu rời rạc bằng MATLAB

## *A. Tín hiệu và hệ thống rời rạc ở miền n*

1.1. Viết chương trình con tạo một dãy thực ngẫu nhiên xuất phát từ n1 đến n2 và có giá trị của biên độ theo phân bố Gauss với trung bình bằng 0, phương sai bằng 1. Yêu cầu chương trình con có các tham số đầu vào và đầu ra được nhập theo câu lệnh với cú pháp:

```
[x,n] = randnseq(n1,n2);
```

***Điền các câu lệnh vào phần trống dưới đây:***

```
function [x,n]=randnseq(n1,n2)
n=[n1:n2];
x=randn(size(n));
```

1.2. Viết chương trình tạo hàm năng lượng của một dãy. Yêu cầu chương trình con có các tham số đầu vào và đầu ra được nhập theo câu lệnh với cú pháp:

```
Ex = energy(x,n);
```

***Điền các câu lệnh vào phần trống dưới đây:***

```
function Ex = energy(x,n)
n = [n1,n2];
Ex = sum(abs(x).^2);
```

1.3. Cho $x(n) = \{1, 2, 3, 4, 5, 6, 7, 6, 5, 4, 3, 2, 1\} - 2 \leq n \leq 10$ Viết chương trình thể hiện trên đồ thị các dãy sau đây:

$a. x_1(n) = 2x(n-5) - 3x(n+4)$

$b. x_2(n) = x(3-n) - x(n)x(n-2)$

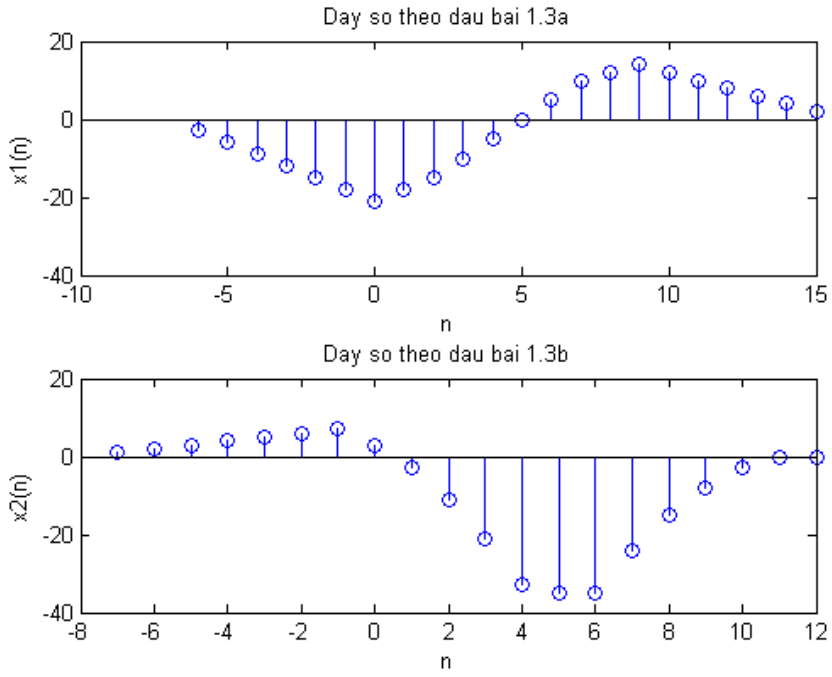
***Điền các câu lệnh vào phần trống dưới đây:***

Các hàm được viết ở file .m riêng

```
% ham dich 1 day tin hieu
function [y,n]=sigshift(x,m,n0)
n=m+n0;
y=x;
% ham dao 1 day tin hieu
function [y,n] = sigfold(x,n)
y=fliplr(x);
n=-fliplr(n);
% ham cong 2 day tin hieu
function [y,n]=sigadd(x1,n1,x2,n2)
n=min(min(n1),min(n2)):max(max(n1),max(n2));
y1=zeros(1,length(n));
y2=y1;
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1;
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;
y=y1+y2;
% ham nhan 2 day tin hieu
function [y,n]=sigmult(x1,n1,x2,n2)
n=min(min(n1),min(n2)):max(max(n1),max(n2));
y1=zeros(1,length(n));
y2=y1;
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1;
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;
y=y1.*y2;
% bat dau lam bai
n = [-2,10];
x = [1:7,6:-1:1];
% ve tin hieu x1[n]
[x11, n11] = sigshift(x, n, 5);
[x12, n12] = sigshift(x, n, -4);
[x1, n1] = sigadd(2 * x11, n11, -3 * x12, n12);
subplot(2,1,1);
stem(n1, x1);
title('Day so theo dau bai 1.3a');
xlabel('n');
ylabel('x1(n)');
% ve tin hieu x2[n]
```

```
[xt, nt] = sigfold(x, n);
[x21, n21] = sigshift(xt, nt, 3);
[xt, nt] = sigshift(x, n, 2);
[x22, n22] = sigmult(x , n, xt , nt);
[x2, n2] = sigadd(x21, n21, -x22, n22);
subplot(2,1,2);
stem(n2, x2);
title('Day so theo dau bai 1.3b');
xlabel('n');
ylabel('x2(n)');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




1.4. Cho hệ thống được mô tả bởi phương trình sai phân tuyến tính hệ số hằng như sau:

$$y(n) - y(n-1) + 0.9y(n-2) = x(n)$$

Sử dụng hàm **filter** của MATLAB, viết chương trình thực hiện các công việc sau:

a. Biểu diễn bằng đồ thị hàm đáp ứng xung đơn vị của hệ thống với -20 ≤n ≤100

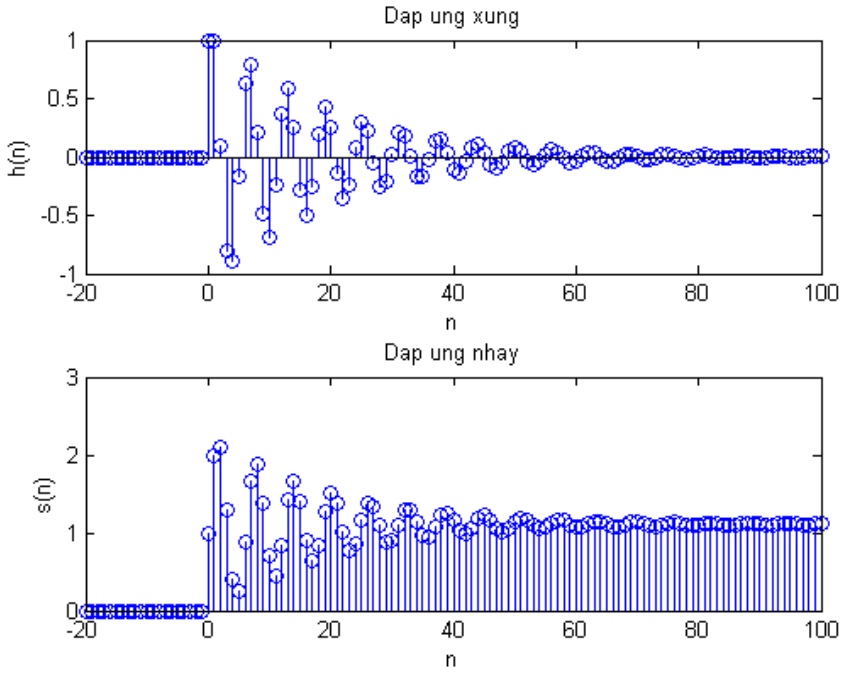
b. Biểu diễn bằng đồ thị dãy đáp ứng của hệ thống với -20 ≤n ≤100 khi dãy đầu vào là dãy nhảy đơn vị.

***Điền các câu lệnh vào phần trống dưới đây:***

Các hàm được viết ở file .m riêng

```
% Day xung don vi
function [x,n]=impseq(n0,n1,n2)
n=[n1:n2];
x=[(n-n0) == 0];
end
% Day nhay don vi
function [x,n] = stepseq(n0,n1,n2)
n = [n1:n2];
x = [(n-n0) >= 0];
end
% Bat dau lam bai
n = -20:100;
b = 1;
a = [1, -1, 0.9];
% dap ung xung
x1 = impseq(0, -20, 100);
h = filter(b, a, x1);
subplot(2, 1, 1);
stem(n , h);
title('Dap ung xung');
xlabel('n');
ylabel('h(n)');
% dap ung nhay
x2 = stepseq(0, -20, 100);
s = filter(b, a, x2);
subplot(2, 1, 2);
stem(n, s);
title('Dap ung nhay');
xlabel('n');
ylabel('s(n)');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




## *B. Tín hiệu và hệ thống rời rạc ở miền Z, miền tần số liên tục ω, và miền tần số rời rạc k*

1.5. Cho dãy $x(n) = 0{,}5^n u(n)$

a. Dựa trên định nghĩa của biến đổi Z, tìm biến đổi Z của dãy trên

b. Kiểm chứng lại kết quả câu a bằng hàm **ztrans**

c. Từ kết quả trên, tìm biến đổi Fourier của x(n)

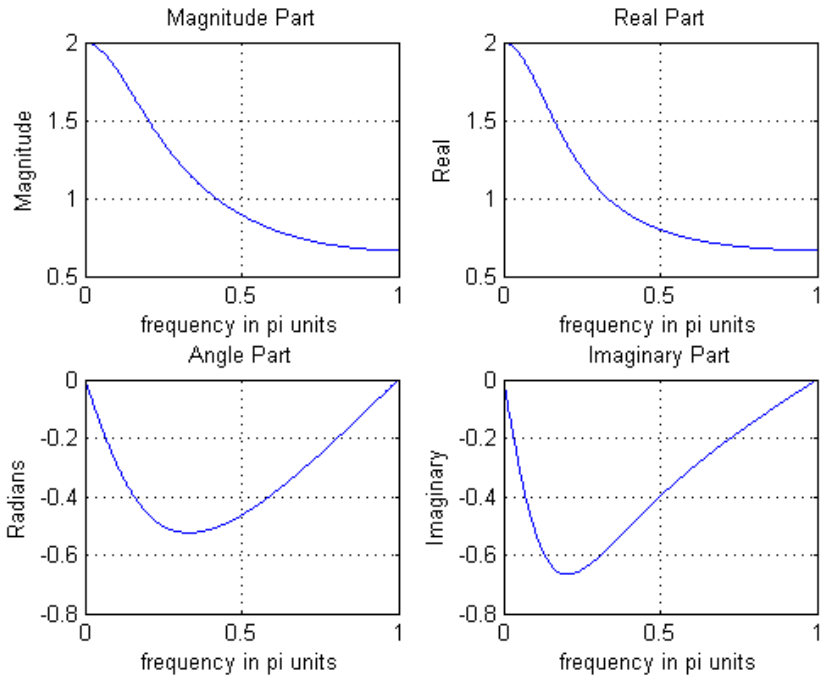
Dùng MATLAB thể hiện trên đồ thị phổ $X(e^{j\omega})$ tại 501 điểm rời rạc trong khoảng [0,π]

***Điền các câu lệnh vào phần trống dưới đây:***

```
w = [0:1:500]*pi/500;
X = exp(j*w) ./ (exp(j*w)- 0.5*ones(1,501));
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
% Bat dau ve do thi
subplot(2,2,1); plot(w/pi,magX); grid;
title('Magnitude Part'); xlabel('frequency in pi units');
ylabel('Magnitude');
```

```
subplot(2,2,3); plot(w/pi,angX); grid;
title('Angle Part'); xlabel('frequency in pi units');
ylabel('Radians');
subplot(2,2,2); plot(w/pi,realX); grid;
title('Real Part'); xlabel('frequency in pi units');
ylabel('Real');
subplot(2,2,4); plot(w/pi,imagX); grid;
title('Imaginary Part'); xlabel('frequency in pi units');
ylabel('Imaginary');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




1.6. Cho dãy x(n) có dạng như sau:

$$x(n) = \{\ldots, 0, 0, \underset{\uparrow}{1}, 2, 3, 4, 5, 0, 0, \ldots\}$$

Đây là một dãy số xác định trong một khoảng hữu hạn từ -1 đến 3.

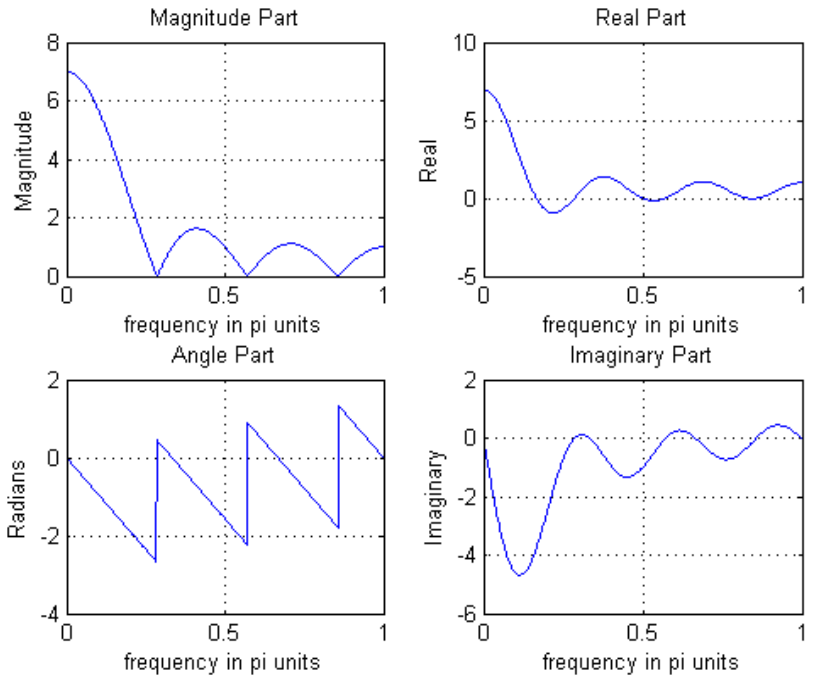
Dựa trên công thức định nghĩa của biến đổi Fourier, viết chương trình tính và thể hiện phổ của dãy x(n) tại 501 điểm rời rạc trong khoảng [0,π].

Cho dãy $x(n) = rect_7(n)$

***Điền các câu lệnh vào phần trống dưới đây:***

```
n = -1:3; x = 1:5;
w = [0:1:500]*pi/500;
% x(n) = rect7(n);
X = (1 - exp(-7j * w)) ./ (1 - exp(-j*w));
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
% Bat dau ve do thi
subplot(2,2,1); plot(w / pi,magX); grid;
title('Magnitude Part'); xlabel('frequency in pi units');
ylabel('Magnitude');
subplot(2,2,3); plot(w / pi,angX); grid;
title('Angle Part'); xlabel('frequency in pi units');
ylabel('Radians');
subplot(2,2,2); plot(w / pi,realX); grid;
title('Real Part'); xlabel('frequency in pi units');
ylabel('Real');
subplot(2,2,4); plot(w / pi,imagX); grid;
title('Imaginary Part'); xlabel('frequency in pi units');
ylabel('Imaginary');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***



1.7. Một hàm ở miền Z được cho với công thức sau đây:

$$X(z) = \frac{z}{3z^2 - 4z + 1}$$

Hàm số X(z) có thể viết dưới dạng tỷ số của hai đa thức theo $z^{-1}$ như sau

$$X(Z) = \frac{z}{3z^2 - 4z + 1} = \frac{z^{-1}}{3 - 4z^{-1} + z^{-2}} = \frac{0 + z^{-1}}{3 - 4z^{-1} + z^{-2}}$$

a. Sử dụng lệnh **residuez** của MATLAB, tính các điểm cực, thặng dư tại các điểm

cực.

b. Từ kết quả câu trên, viết công thức khai triển X(z) thành tổng các phân thức đơn giản, từ đó tìm biến đổi Z ngược của X(z), cho biết x(n) là một dãy nhân quả.

c. Kiểm chứng lại kết quả câu b bằng hàm **iztrans**

***Điền các câu lệnh vào phần trống dưới đây:***

```
b = [0 1]; a = [3 -4 1];
[R,p,C] = residuez(b,a)
% Tim lai phan thuc
[b a] = residuez(R,p,C)
```

1.8. Cho hàm X(z) với công thức như sau:

$$X(z) = \frac{1}{(1 - 0{,}9z^{-1})^2(1 - 0{,}9z^{-1})}$$

a. Viết chương trình tính các điểm cực, thặng dư của các điểm cực của hàm X(z) trên (gợi ý: có thể dùng hàm poly của MATLAB để khôi phục lại đa thức mẫu số từ một mảng các nghiệm của đa thức - mảng các điểm cực của X(z))

b. Từ kết quả câu trên, viết công thức khai triển X(z) thành tổng các phân thức đơn giản, từ đó tìm biến đổi Z ngược của X(z) trên miền $|z| > 0{,}9$

***Điền các câu lệnh vào phần trống dưới đây:***

```
b = [1]
a = poly([0.9 0.9 0.9])
[R,p,C] = residuez(b,a)
% Tim lai phan thuc
[b a] = residuez(R,p,C)
```

1.9. Cho hệ thống nhân quả biểu diễn bởi phương trình sau:

$$y(n) - 0{,}9y(n-1) = x(n)$$

a. Tìm hàm truyền đạt của hệ thống

Sau đó thực hiện các công việc sau:

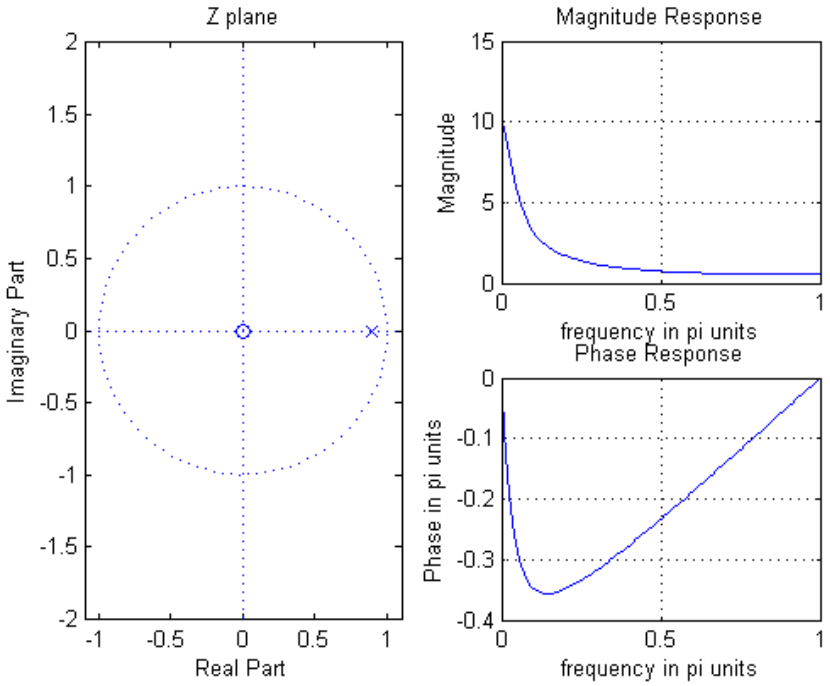
b. Dùng lệnh **zplane** của MATLAB biểu diễn trên đồ thị mặt phẳng Z sự phân bố các điểm cực và điểm không

c. Tính và biểu diễn trên đồ thị hàm đáp ứng tần số $H(e^{j\omega})$của hệ thống (bao gồm đáp ứng biên độ- tần số và đáp ứng pha - tần số) tại 200 điểm rời rạc trên đường tròn đơn vị

***Điền các câu lệnh vào phần trống dưới đây:***

```
b = [1 0]; a = [1 -0.9];
% Tim phan bo diem cuc va diem khong
subplot(1,2,1);
zplane(b,a);
title('Z plane');
% Tim dap ung tan so bang cach danh gia 200 diem roi rac
% cua H(z) tren duong tron don vi
[H, w] = freqz(b,a,200,'whole');
magH = abs(H(1:101)); phaH= angle(H(1:101));
% Ve dap ung tan so
subplot(2,2,2); plot(w(1:101)/pi,magH); grid;
title('Magnitude Response');
xlabel('frequency in pi units');
ylabel('Magnitude');
subplot(2,2,4); plot(w(1:101)/pi,phaH/pi); grid;
title('Phase Response');
xlabel('frequency in pi units');
ylabel('Phase in pi units');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




1.10. Tạo các hàm thực hiện việc biến đổi Fourier rời rạc thuận (đặt tên là hàm **dft**) và Fourier rời rạc ngược (đặt tên là hàm **idft**). Dựa trên các hàm **dft** được xây dựng ở trên, tìm biến đổi Fourier rời rạc của dãy có chiều dài N=20:

$$x(n) = \begin{cases} 1 & 0 \le n \le 4 \\ 0 & n \text{ còn lại} \end{cases}$$

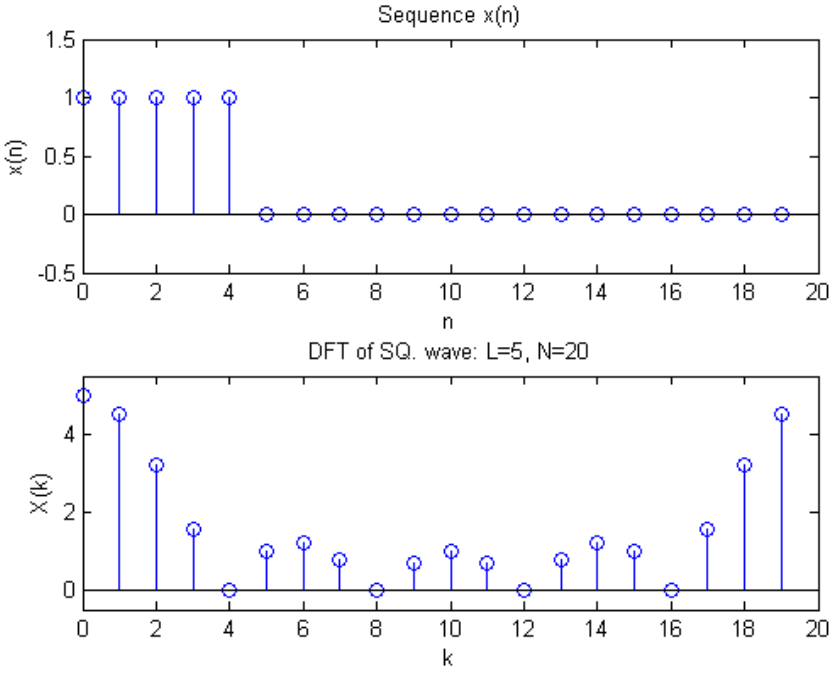
***Điền các câu lệnh vào phần trống dưới đây:***

Các hàm được viết trong file .m riêng

```
% Ham bien doi Fourier roi rac thuan
function [Xk] = dft(xn,N)
n = [0:1:N-1];
k = [0:1:N-1];
WN = exp(-j*2*pi/N);
nk = n' * k;
WNnk = WN .^ nk; % ma tran DFT
Xk = xn * WNnk;
end
% Ham bien doi Fourier roi rac nguoc
function [xn] = idft(Xk,N)
n = [0:1:N-1];
k = [0:1:N-1];
```

```
WN = exp(-j*2*pi/N);
nk = n' * k;
WNnk = WN .^ (-nk); % ma tran IDFT
xn = (Xk * WNnk)/N;
end
% Bat dau lam bai
L = 5; N = 20;
n = [0:N-1];
xn = [ones(1,L), zeros(1,N-L)];
k = n;
Xk = dft(xn,N);
magXk = abs(Xk);
% Bat dau ve do thi
subplot(2,1,1); stem(n,xn);
axis([min(n),max(n)+1,-0.5,1.5]);
title('Sequence x(n)');
xlabel('n'); ylabel('x(n)');
subplot(2,1,2); stem(k,magXk);
axis([min(k),max(k)+1,-0.5,5.5]);
title('DFT of SQ. wave: L=5, N=20');
xlabel('k'); ylabel('X(k)');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***



# BÀI 2. Thiết kế bộ lọc số bằng MATLAB

## *A. Thiết kế bộ lọc có đáp ứng xung chiều dài hữu hạn (bộ lọc số FIR)*

Để tổng hợp một bộ lọc FIR, các tham số đầu vào được cho với các ký hiệu như sau

- Tần số cắt dải thông $\omega_p$
- Tần số cắt dải thông $\omega_s$
- Bề rộng dải quá độ $\Delta\omega$
- Độ gợn sóng dải thông $\delta_1$
- Độ gợn sóng dải chắn $\delta_2$

Ngoài ra các tham số được cho theo đơn vị decibel như sau:

- Độ gợn sóng dải thông và độ suy giảm dải chắn theo dB, được tính bằng công thức:

$$R_p = -20log\frac{1-\delta_1}{1+\delta_1}[dB] \qquad A_s = -20log\frac{\delta_2}{1+\delta_1}[dB]$$

**Các bước thực hành**

2.1. Tạo các hàm thể hiện độ lớn của đáp ứng tần số các bộ lọc FIR loại 1 từ dãy đáp ứng xung của chúng theo chương trình mẫu bằng cách gõ các dòng lệnh cho ở bảng dưới đây vào cửa sổ soạn thảo (Editor) và ghi lại theo tên tệp là **Hr_Type1.m**:

*Hàm độ lớn của đáp ứng tần số bộ lọc FIR loại 1:*

```
function [Hr,w,a,L] = Hr_Type1(h)
M = length(h);
L = (M-1)/2;
a = [h(L+1) 2*h(L:-1:1)];
n = [0:1:L];
w = [0:1:500]'*pi/500;
```

```
Hr = cos(w*n)*a';
```

2.2. Viết chương trình tính hàm độlớn của đáp ứng tần sốbộlọc FIR loại 2, FIR loại 3 và bộ lọc FIR loại 4 với các tham số đầu vào và đầu ra được nhập theo các câu lệnh:

`>> [Hr,w,b,L] = Hr_Type2(h)` -> cho bộ lọc FIR loại 2

***Điền các câu lệnh vào phần trống dưới đây:***

```
function [Hr,w,b,L] = Hr_Type2(h)
M = length(h);
L = M/2;
b = 2*h(L:-1:1);
n = [1:1:L]; n = n-0.5;
w = [0:1:500]'*pi/500;
Hr = cos(w*n)*b';
```

`>> [Hr,w,c,L] = Hr_Type3(h)` -> cho bộ lọc FIR loại 3

***Điền các câu lệnh vào phần trống dưới đây:***

```
function [Hr,w,c,L] = Hr_Type3(h)
M = length(h);
L = (M-1)/2;
c = 2*h(L:-1:1);
n = [1:1:L];
w = [0:1:500]'*pi/500;
Hr = sin(w*n)*c';
```

`>> [Hr,w,d,L] = Hr_Type4(h)` -> cho bộ lọc FIR loại 4

***Điền các câu lệnh vào phần trống dưới đây:***

```
function [Hr,w,d,L] = Hr_Type4(h)
M = length(h);
L = M/2;
d = 2*h(L:-1:1);
n = [1:1:L]; n = n-0.5;
w = [0:1:500]'*pi/500;
Hr = sin(w*n)*d';
```

2.3. Cho bộ lọc FIR với đáp ứng xung như sau:

$$h(n) = \{-4, 1, -1, -2, \underset{\uparrow}{5}, 6, 5, -1, -2, 1, -4\}$$

a. Xác định loại của bộ lọc.

Tính và biểu diễn trên đồ thị:

b. Dãy đáp ứng xung của bộ lọc

c. Các hệ số của bộ lọc

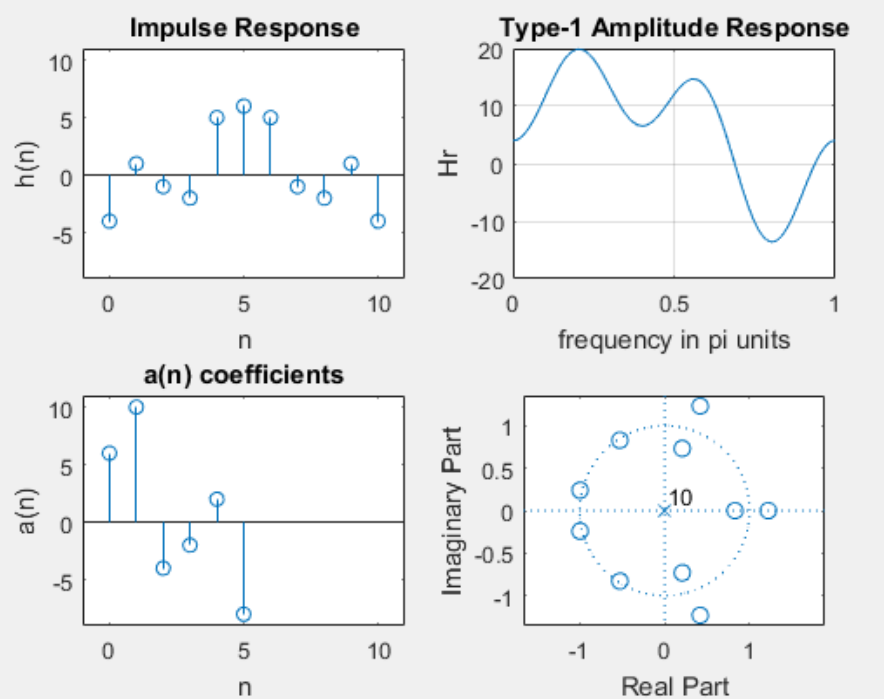
d. Hàm độ lớn của đáp ứng tần số

e. Phân bố điểm cực và điểm không

***Điền các câu lệnh vào phần trống dưới đây:***

```
h = [-4,1,-1,-2,5,6,5,-1,-2,1,-4];
M = length(h); n =0:M-1;
[Hr,w,a,L] = Hr_Type1(h);
a, L
amax = max(a)+1; amin = min(a)-1;
%
subplot(2,2,1); stem(n,h);
axis([-1,2*L+1,amin,amax]);
title('Impulse Response');
xlabel('n'); ylabel('h(n)');
%
subplot(2,2,3); stem(0:L,a);
axis([-1,2*L+1,amin,amax]);
title('a(n) coefficients');
xlabel('n'); ylabel('a(n)');
%
subplot(2,2,2); plot(w/pi,Hr); grid;
title('Type-1 Amplitude Response');
xlabel('frequency in pi units'); ylabel('Hr');
%
subplot(2,2,4); zplane(h,1);
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.4. Cho bộ lọc FIR với đáp ứng xung như sau:

$$h(n) = \{-4, 1, -1, -2, \underset{\uparrow}{5}, 6, -6, -5, 1, 2, -1, 4\}$$

a. Xác định loại của bộ lọc.

Tính và biểu diễn trên đồ thị:

b. Dãy đáp ứng xung của bộ lọc

c. Các hệ số của bộ lọc

d. Hàm độ lớn của đáp ứng tần số

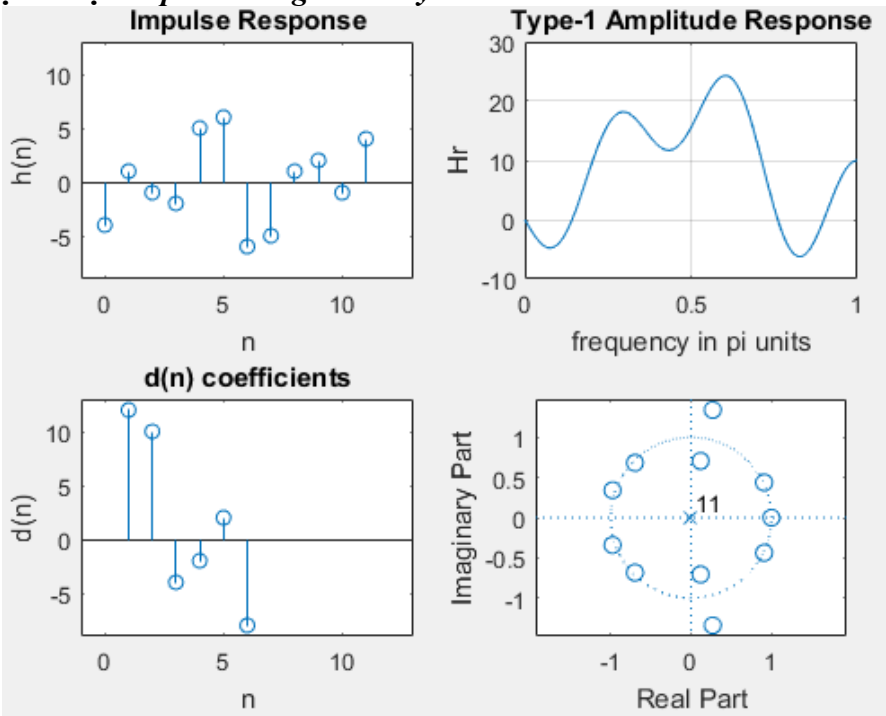
e. Phân bố điểm cực và điểm không

***Điền các câu lệnh vào phần trống dưới đây:***

```
h = [-4,1,-1,-2,5,6,-6,-5,1,2,-1,4];
M = length(h); n =0:M-1;
[Hr,w,d,L] = Hr_Type4(h);
d, L
amax = max(d)+1; amin = min(d)-1;
%
subplot(2,2,1); stem(n,h);
```

```
axis([-1,2*L+1,amin,amax]);
title('Impulse Response');
xlabel('n'); ylabel('h(n)');
%
subplot(2,2,3); stem(1:L,d);
axis([-1,2*L+1,amin,amax]);
title('d(n) coefficients');
xlabel('n'); ylabel('d(n)');
%
subplot(2,2,2); plot(w/pi,Hr); grid;
title('Type-1 Amplitude Response');
xlabel('frequency in pi units'); ylabel('Hr');
%
subplot(2,2,4); zplane(h,1);
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.5. Thiết kế bộ lọc thông thấp theo phương pháp cửa số với các tham số đầu vào như sau:

$\omega_p = 0{,}2\pi,$ $R_p = 0{,}25dB$

$\omega_s = 0{,}3\pi,$ $A_s = 50dB$

Tính và biểu diễn trên đồ thị:

a. Dãy đáp ứng xung của bộ lọc lý tưởng

b. Dãy hàm cửa sổ Hamming

c. Hàm độ lớn tuyệt đối của đáp ứng tần số

d. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

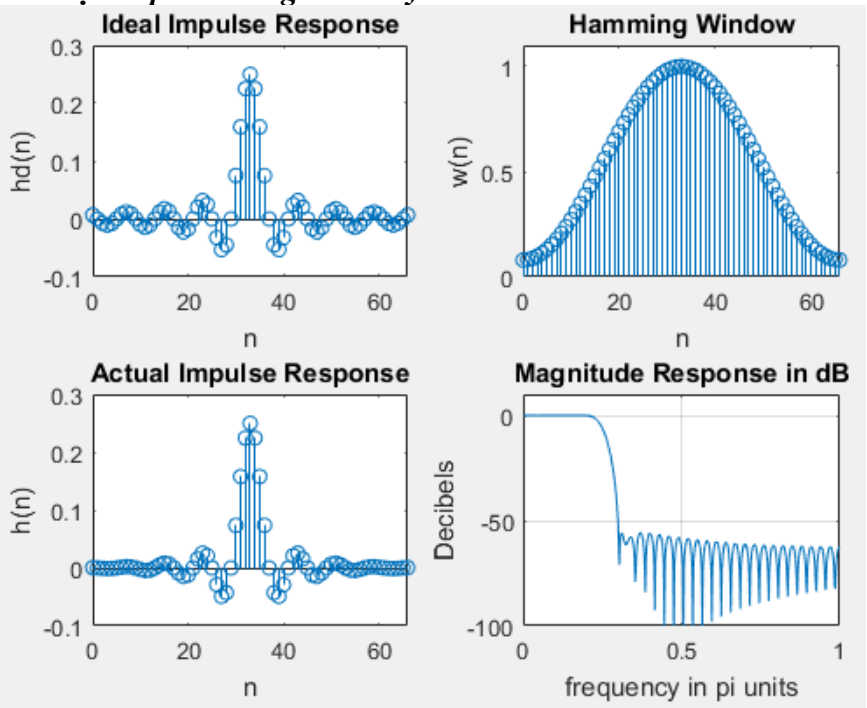
***Điền các câu lệnh vào phần trống dưới đây:***

Các hàm được viết trong file .m riêng

```
% Dap ung xung bo loc thong thap li tuong
function hd = ideal_lp(wc,M)
alpha = (M-1)/2;
n = [0:1:(M-1)];
m = n - alpha + eps;
hd = sin(wc*m) ./ (pi*m);
end
% Dap ung tan so he thong so
function [db,mag,pha,grd,w] = freqz_m(b,a)
[H,w] = freqz(b,a,1000,'whole');
H = (H(1:1:501))';
w = (w(1:1:501))';
mag = abs(H);
db = 20*log10((mag+eps)/max(mag));
pha = angle(H);
grd = grpdelay(b,a,w);
end
% Bat dau lam bai
wp = 0.2*pi; ws =0.3*pi;
tr_width = ws - wp;
M = ceil(6.6*pi/tr_width) + 1
n = [0:1:M-1];
wc = (ws+wp)/2;
hd = ideal_lp(wc,M);
w_ham = (hamming(M))';
h = hd .* w_ham;
[db,mag,pha,grd,w] = freqz_m(h,[1]);
delta_w = 2*pi/1000;
Rp = -(min(db(1:1:wp/delta_w+1)))
As = -round(max(db(ws/delta_w+1:1:501)))
%plot
subplot(2,2,1); stem(n,hd);
axis([0,M-1,-0.1,0.3]);
title('Ideal Impulse Response');
xlabel('n'); ylabel('hd(n)');
%
subplot(2,2,2); stem(n,w_ham);
```

```
axis([0,M-1,0,1.1]);
title('Hamming Window');
xlabel('n'); ylabel('w(n)');
%
subplot(2,2,3); stem(n,h);
axis([0,M-1,-0.1,0.3]);
title('Actual Impulse Response');
xlabel('n'); ylabel('h(n)');
%
subplot(2,2,4); plot(w/pi,db); grid;
axis([0,1,-100,10]);
title('Magnitude Response in dB');
xlabel('frequency in pi units'); ylabel('Decibels');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.6. Thiết kế bộ lọc thông thấp theo phương pháp lấy mẫu tần số với các tham số đầu vào như sau:

$\omega_p = 0{,}2\pi,$ $R_p = 0{,}25dB$

$\omega_s = 0{,}3\pi,$ $A_s = 50dB$

Giả sử rằng ta chọn đáp ứng xung có chiều dài 60 tương đương với lấy 60 mẫu tần số trong khoảng [0,2π). Dải thông có độ rộng là 0,2π tương đương với 7 mẫu nhận giá trị 1.

Giả sử tiếp rằng quá trình tối ưu hoá chỉ ra nên chọn dải chuyển tiếp 2 mẫu nhận các giá trị T1 = 0,5925 và T2 = 0,1099. Vậy dãy mẫu các tần số được cho như sau:

$$h(n) = \left\{1,1,1,1,1,1,1,T_1,T_2,\underbrace{0,\ldots,0}_{43\ mẫu\ 0}\ T_2,T_1,1,1,1,1,1,1,1\right\}$$

Tính và biểu diễn trên đồ thị:

a. Dãy các mẫu tần số

b. Dãy đáp ứng xung của bộlọc thực tế

c. Hàm độ lớn tuyệt đối của đáp ứng tần số

d. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

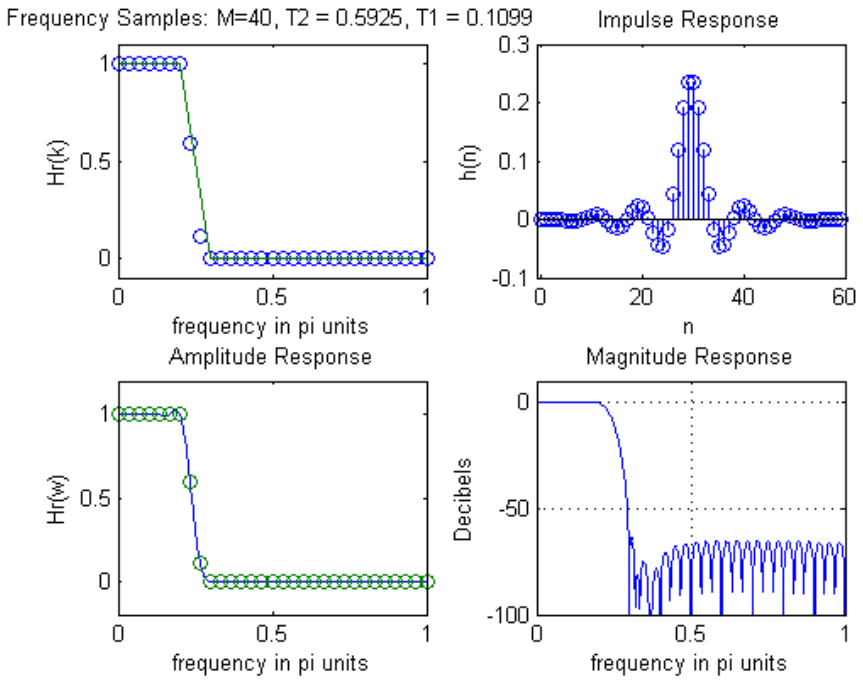
***Điền các câu lệnh vào phần trống dưới đây:***

Bài này có dùng hàm freqz_m và Hr_Type 2 ở phần trên

```
M = 60; alpha = (M-1)/2; l = 0:M-1; wl = (2*pi/M)*l;
Hrs =
[ones(1,7),0.5925,0.1099,zeros(1,43),0.1099,0.5925,ones(1,6)];
% Day dap ung tan so mau ly tuong
Hdr = [1,1,0,0]; wdl = [0,0.2,0.3,1];
% Dap ung tan so ly tuong de bieu dien do thi
k1 = 0:floor((M-1)/2); k2 = floor((M-1)/2)+1:M-1;
angH = [-alpha*(2*pi)/M*k1, alpha*(2*pi)/M*(M-k2)];
H = Hrs.*exp(j*angH);
h = real(ifft(H,M));
[db,mag,pha,grd,w] = freqz_m(h,1);
[Hr,ww,a,L] = Hr_Type2(h);
%plot
subplot(2,2,1); plot(wl(1:31)/pi,Hrs(1:31),'o',wdl,Hdr);
axis([0,1,-0.1,1.1]);
title('Frequency Samples: M=40, T2 = 0.5925, T1 = 0.1099');
xlabel('frequency in pi units'); ylabel('Hr(k)');
%
subplot(2,2,2); stem(l,h);
axis([-1,M,-0.1,0.3]);
title('Impulse Response');
```

```
xlabel('n'); ylabel('h(n)');
%
subplot(2,2,3); plot(ww/pi,Hr,wl(1:31)/pi,Hrs(1:31),'o');
axis([0,1,-0.2,1.2]);
title('Amplitude Response');
xlabel('frequency in pi units'); ylabel('Hr(w)');
%
subplot(2,2,4); plot(w/pi,db);
axis([0,1,-100,10]); grid
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Decibels');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.7. Thiết kế bộ lọc thông thấp theo phương pháp lặp (thuật toán của Parks và McClellan) với các tham số đầu vào như sau:

$\omega_p = 0{,}2\pi,$ $R_p = 0{,}25dB$

$\omega_s = 0{,}3\pi,$ $A_s = 50dB$

Trước tiên xuất phát từ độ dài của dãy đáp ứng M theo công thức

$$M = \frac{-20log\sqrt{\delta_1\delta_2}-13}{14{,}6\Delta f}, \text{ với } \Delta f = \frac{\omega_s-\omega_p}{2\pi}$$

Lặp công việc tìm bộ lọc tối ưu theo nghĩa Chebyshev (dùng lệnh **firpm**) và tăng M sau mỗi lần lặp để tìm ra bộ lọc thoả mãn yêu cầu thiết kế, sau đó tính và biểu diễn trên đồ thị:

a. Dãy đáp ứng xung của bộ lọc thực tế

b. Hàm độ lớn tuyệt đối của đáp ứng tần số

c. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

d. Hàm sai số $E(\omega)$

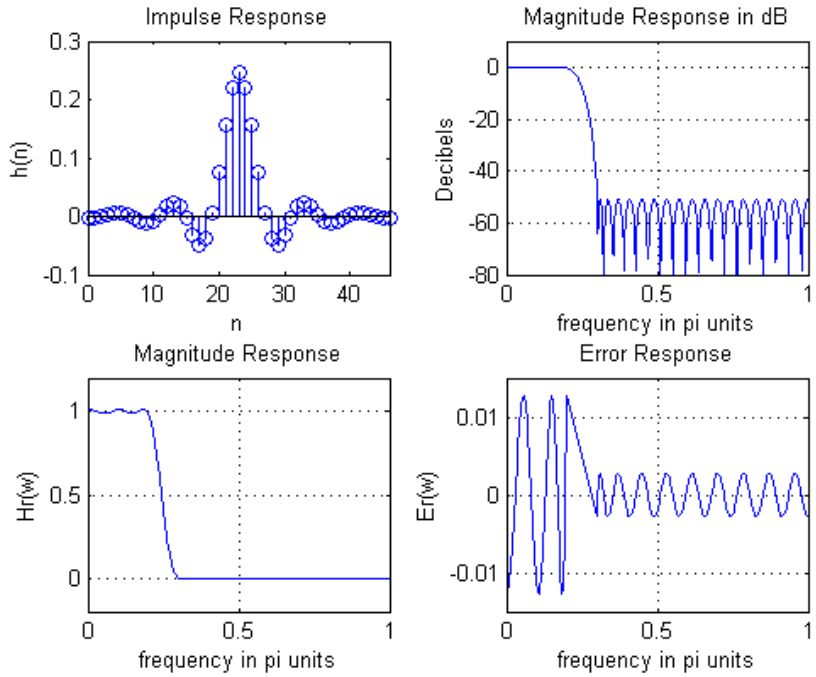
***Điền các câu lệnh vào phần trống dưới đây:***

Bài này có dùng hàm freqz_m ở phần trên

```
wp = 0.2*pi; ws =0.3*pi; Rp = 0.25; As = 50;
delta_w = 2*pi/1000;
wsi = ws/delta_w+1;
delta1 = (10^(Rp/20)-1)/(10^(Rp/20)+1);
delta2 = (1+delta1)*(10^(-As/20));
deltaH = max(delta1,delta2);
deltaL = min(delta1,delta2);
weights = [delta2/delta1 1];
deltaf = (ws-wp)/(2*pi);
M = ceil((-20*log10(sqrt(delta1*delta2))-13)/(14.6*deltaf)+1)
f = [0 wp/pi ws/pi 1];
m = [1 1 0 0];
h = firpm(M-1,f,m,weights);
[db,mag,pha,grd,w] = freqz_m(h,[1]);
Asd = -max(db(wsi:1:501))
while Asd<As
M = M+1
[h,ERR,RES] = firpm(M-1,f,m,weights);
[db,mag,pha,grd,w] = freqz_m(h,[1]);
Asd = -max(db(wsi:1:501))
end
%plot
n = [0:1:M-1];
subplot(2,2,1); stem(n,h);
axis([0,M-1,-0.1,0.3]);
title('Impulse Response');
xlabel('n'); ylabel('h(n)');
%
subplot(2,2,2); plot(w/pi,db); grid;
axis([0,1,-80,10]);
title('Magnitude Response in dB');
xlabel('frequency in pi units'); ylabel('Decibels');
```

```
%
subplot(2,2,3); plot(w/pi,mag); grid;
axis([0,1,-0.2,1.2]);
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Hr(w)');
%
subplot(2,2,4); plot(RES.fgrid,RES.error); grid;
axis([0,1,-0.0150,0.0150]);
title('Error Response');
xlabel('frequency in pi units'); ylabel('Er(w)');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




## *B. Thiết kế bộ lọc có đáp ứng xung chiều dài vô hạn (bộ lọc số IIR)*

**Các bước thực hành**

2.8. Thiết kế bộ lọc thông thấp tương tự, định dạng Chebyshev-I, cửa số với các tham số đầu vào như sau:

$\omega_p = 0{,}2\pi,$ $R_p = 1dB$

$\omega_s = 0{,}3\pi,$ $A_s = 16dB$

Viết chương trình tính và biểu diễn trên đồthị:

a. Độ lớn của đáp ứng tần số

b. Hàm đáp ứng pha của bộ lọc

c. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

d. Hàm đáp ứng xung của bộ lọc tương tự

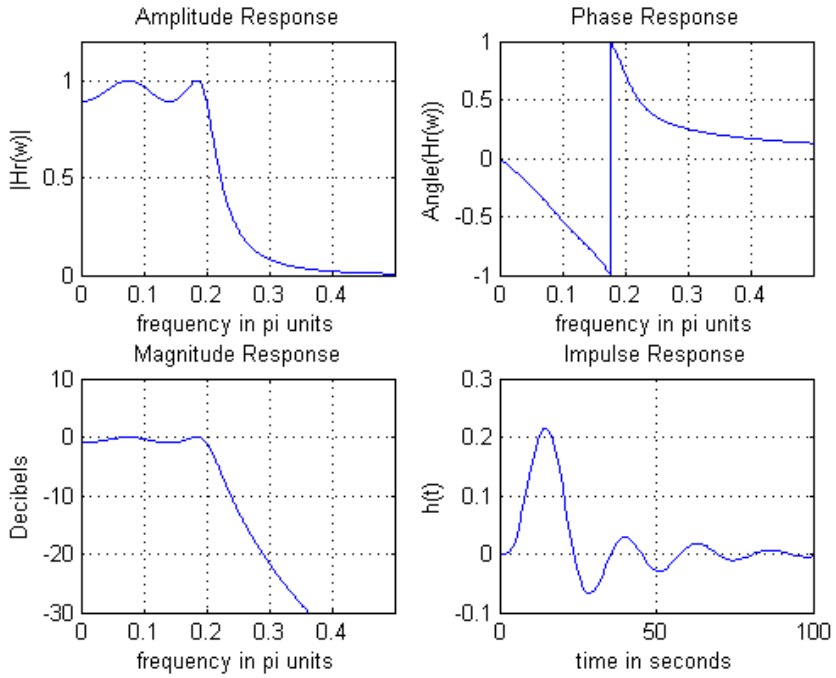
***Điền các câu lệnh vào phần trống dưới đây:***

Các hàm được viết trong file .m riêng

```
% Ham tinh dap ung tan so
function [db,mag,pha,w] = freqs_m(b,a,wmax);
w = [0:1:500]*wmax/500;
H = freqs(b,a,w);
mag = abs(H);
db = 20*log10((mag+eps)/max(mag));
pha = angle(H);
end
% Ham tra ve he so cac da thuc tu va da thuc mau cua ham truyen
dat cho thiet ke bo loc dang Chebyshev I co tan so cat tuy y theo
chuong trinh mau
function [b,a] = u_chb1ap(N,Rp,Omegac)
[z,p,k] = cheb1ap(N,Rp);
a = real(poly(p));
aNn = a(N+1);
p = p*Omegac;
a = real(poly(p));
aNu = a(N+1);
k = k*aNu/aNn;
B = real(poly(z));
b0 = k;
b = k*B;
end
% Ham tra ve thiet ke bo loc thong thap tuong tu, đinh dang
Chebyshev co bac toi uu theo chuong trinh mau
function [b,a] = afd_chb1(Wp,Ws,Rp,As)
if Wp <= 0
error('Tan so cat dai thong phai lon hon 0')
end
if Ws <= Wp
error('Tan so cat dai thong phai lon hon tan so cat dai chan')
end
if (Rp <= 0) | (As<0)
```

```
error('Do gon dai thong va/hoac Do suy giam dai chan phai lon hon
0')
end
%
ep = sqrt(10^(Rp/10)-1);
A = 10^(As/20);
OmegaC = Wp;
OmegaR = Ws/Wp;
g = sqrt(A*A-1)/ep;
N=ceil(log10(g+sqrt(g*g-1)) /log10(OmegaR+sqrt(OmegaR*OmegaR-1)));
fprintf('\n*** Bac cua bo loc Chebyshev-1 = %2.0f\n',N);
[b,a] = u_chb1ap(N,Rp,OmegaC);
End
%Bat dau lam bai
% Chi tieu ky thuat cua bo loc tuong tu: Chebyshev-I
wp =0.2*pi; % digital Passband freq in Hz
ws =0.3*pi; % digital Stopband freq in Hz
Rp = 1; % Passband ripple in dB
As = 15; % Stopband attenuation in dB
% Tinh toan bo loc tuong tu:
[b,a] = afd_chb1(wp,ws,Rp,As)
%
[db,mag,pha,w] = freqs_m(b,a,pi/2);
[h,x,t] = impulse(b,a)
%plot
figure(37); clf;
%
subplot(2,2,1); plot(w/pi,mag);
axis([0,0.5,0,1.2]); grid
title('Amplitude Response');
xlabel('frequency in pi units'); ylabel('|Hr(w)|');
%
subplot(2,2,3); plot(w/pi,db);
axis([0,0.5,-30,10]); grid
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Decibels');
%
subplot(2,2,2); plot(w/pi,pha/pi);
axis([0,0.5,-1,1]); grid
title('Phase Response');
xlabel('frequency in pi units'); ylabel('Angle(Hr(w))');
%
subplot(2,2,4); plot(h);
axis([0,100,-0.1,0.3]); grid
title('Impulse Response');
xlabel('time in seconds'); ylabel('h(t)');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.9. Chuyển đổi bộ lọc với các tham số đã cho ở phần 2.8 sang bộ lọc số bằng phương pháp biến đổi song tuyến. Hàm **bilinear** cho phép thực hiện việc chuyển đổi này.

Tính và biểu diễn trên đồ thị:

a. Độ lớn của đáp ứng tần số

b. Hàm đáp ứng pha của bộ lọc

c. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

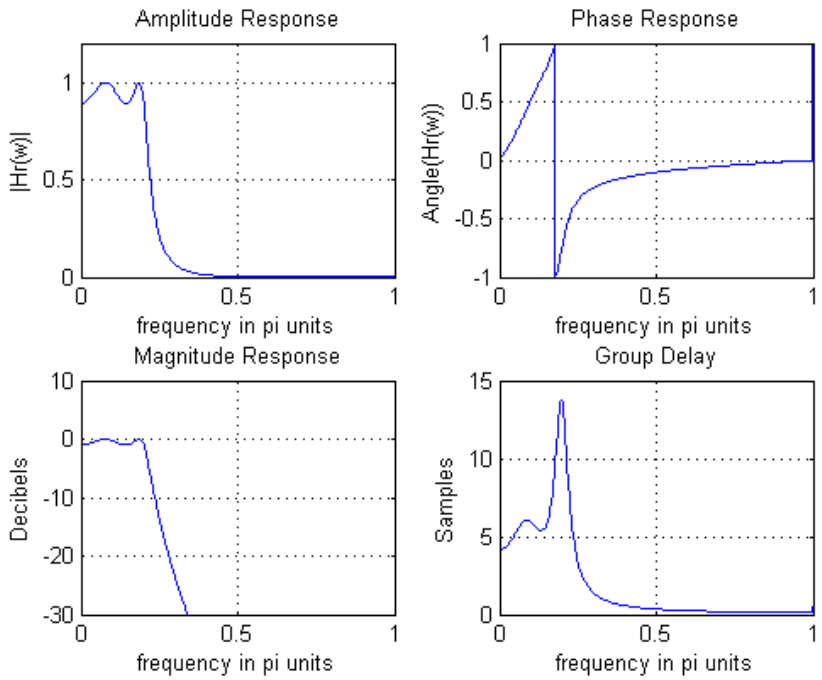
d. Trễ nhóm theo tần số.

***Điền các câu lệnh vào phần trống dưới đây:***

Bài này có dùng hàm afd_chb1, u_chb1ap, freqz_m ở phần trên

```
% Chi tieu ky thuat cua bo loc so:
wp =0.2*pi; % digital Passband freq in Hz
ws =0.3*pi; % digital Stopband freq in Hz
Rp = 1; % Passband ripple in dB
As = 15; % Stopband attenuation in dB
% Chi tieu ky thuat cua bo loc tuong tu: Anh xa nguoc
T = 1; Fs =1/T; % Dat T=1
OmegaP = (2/T)*tan(wp/2);
OmegaS = (2/T)*tan(ws/2);
% Tinh toan bo loc tuong tu:
[cs, ds] = afd_chb1(OmegaP,OmegaS,Rp,As);
% Bien doi song tuyen:
[b,a] = bilinear(cs,ds,Fs);
%
[db,mag,pha,grd,w] = freqz_m(b,a);
%plot
figure(37); clf;
%
subplot(2,2,1); plot(w/pi,mag);
axis([0,1,0,1.2]); grid
title('Amplitude Response');
xlabel('frequency in pi units'); ylabel('|Hr(w)|');
%
subplot(2,2,3); plot(w/pi,db);
axis([0,1,-30,10]); grid
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Decibels');
%
subplot(2,2,2); plot(w/pi,pha/pi);
axis([0,1,-1,1]); grid
title('Phase Response');
xlabel('frequency in pi units'); ylabel('Angle(Hr(w))');
%
subplot(2,2,4); plot(w/pi,grd);
axis([0,1,0,15]); grid
title('Group Delay');
xlabel('frequency in pi units'); ylabel('Samples');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***




2.10. Thực hiện yêu cầu của câu 2.9 theo phương pháp bất biến xung, dùng hàm **impinvar** của MATLAB. So sánh kết quả thu được với câu trên.

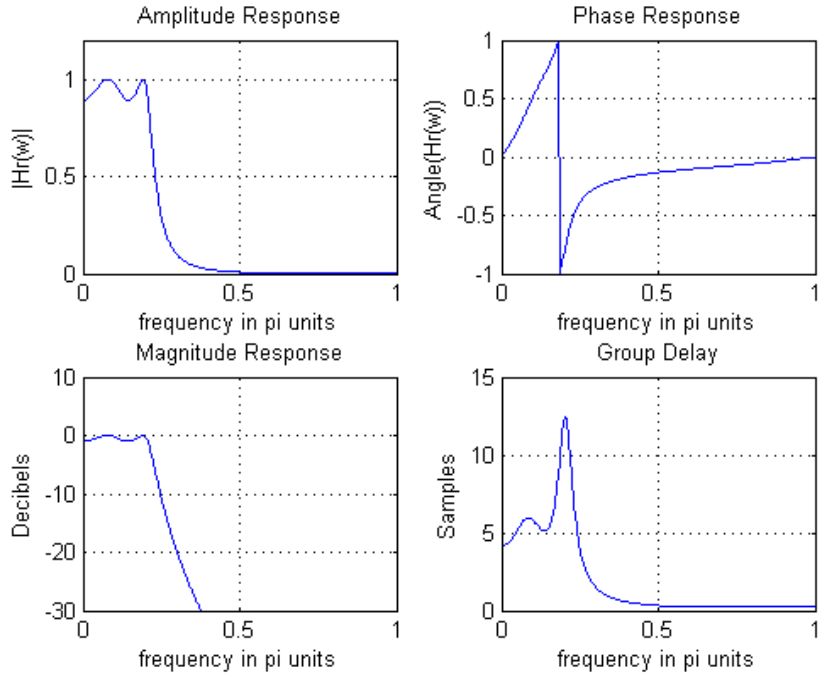
***Điền các câu lệnh vào phần trống dưới đây:***

Bài này có dùng hàm afd_chb1, u_chb1ap, freqz_m ở phần trên

```
% Chi tieu ky thuat cua bo loc so:
wp =0.2*pi; % digital Passband freq in Hz
ws =0.3*pi; % digital Stopband freq in Hz
Rp = 1; % Passband ripple in dB
As = 15; % Stopband attenuation in dB
% Chi tieu ky thuat cua bo loc tuong tu: Anh xa nguoc
T = 1; Fs =1/T; % Dat T=1
OmegaP = (2/T)*tan(wp/2);
OmegaS = (2/T)*tan(ws/2);
% Tinh toan bo loc tuong tu:
[cs, ds] = afd_chb1(OmegaP,OmegaS,Rp,As);
% Bien doi song tuyen:
[b,a] = impinvar(cs,ds,Fs);
%
[db,mag,pha,grd,w] = freqz_m(b,a);
%plot
figure(37); clf;
%
```

```
subplot(2,2,1); plot(w/pi,mag);
axis([0,1,0,1.2]); grid
title('Amplitude Response');
xlabel('frequency in pi units'); ylabel('|Hr(w)|');
%
subplot(2,2,3); plot(w/pi,db);
axis([0,1,-30,10]); grid
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Decibels');
%
subplot(2,2,2); plot(w/pi,pha/pi);
axis([0,1,-1,1]); grid
title('Phase Response');
xlabel('frequency in pi units'); ylabel('Angle(Hr(w))');
%
subplot(2,2,4); plot(w/pi,grd);
axis([0,1,0,15]); grid
title('Group Delay');
xlabel('frequency in pi units'); ylabel('Samples');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***



2.11. Tạo hàm thực hiện việc chuyển đổi băng tần số, trả về hàm truyền đạt của bộ lọc mới với tham số đầu vào là hàm truyền đạt của bộ lọc thông thấp, hàm đa thức thể hiện phép đổi biến số độc lập, ghi lại theo tên tệp là **zmapping.m**:

***Điền các câu lệnh vào phần trống dưới đây:***

```
function [bz,az] = zmapping(bZ,aZ,Nz,Dz)
% Chuyen doi bang tan so tu mien Z sang mien z
% ------------------------------------------------------
% [bz,az] = zmapping(bZ,aZ,Nz,Dz)
% perform:
% b(z)   b(Z)|
% ---- = ----|    N(z)
% a(z)   a(Z)|Z = ----
%                 D(z)
%
bzord = (length(bZ)-1)*(length(Nz)-1);
azord = (length(aZ)-1)*(length(Dz)-1);
bz = zeros(1,bzord+1);
for k = 0:bzord
pln = [1];
for l = 0:k-1
pln = conv(pln,Nz);
end
pld = [1];
for l = 0:bzord-k-1
pld = conv(pld,Dz);
end
bz = bz+bZ(k+1)*conv(pln,pld);
end
%
az = zeros(1,azord+1);
for k = 0:azord
pln = [1];
for l = 0:k-1
pln = conv(pln,Nz);
end
pld = [1];
for l = 0:azord-k-1
pld = conv(pld,Dz);
end
az = az+aZ(k+1)*conv(pln,pld);
end
%
az1 = az(1); az = az/az1; bz=bz/az1;
```

2.12. Viết chương trình chuyển đổi từ bộ lọc thông thấp theo thiết kế của câu 1.9 sang bộ lọc thông cao có tần số cắt $\omega_c=0,6\pi$. Tính và biểu diễn trên đồ thị

a. Độ lớn của đáp ứng tần số

b. Hàm đáp ứng pha của bộ lọc

c. Hàm độ lớn tương đối tính theo dB của đáp ứng tần số

d. Trễ nhóm theo tần số.

***Điền các câu lệnh vào phần trống dưới đây:***

Bài này có dùng hàm zmapping, afd_chb1, u_chb1ap, freqz_m ở phần trên

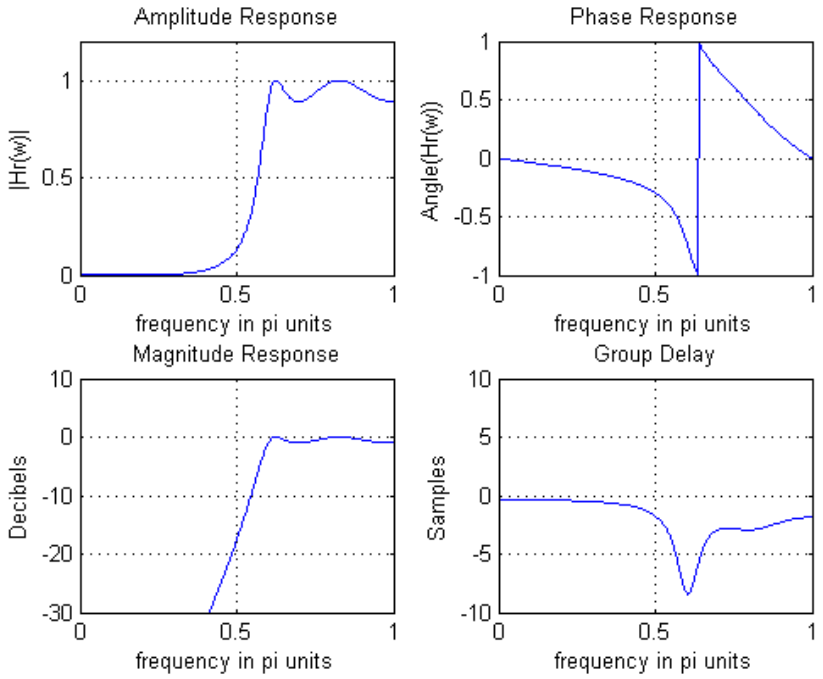
```
% Chi tieu ky thuat cua bo loc so:
wpl =0.2*pi; % digital Passband freq in Hz
wsl =0.3*pi; % digital Stopband freq in Hz
Rp = 1; % Passband ripple in dB
As = 15; % Stopband attenuation in dB
% Chi tieu ky thuat cua bo loc tuong tu: Anh xa nguoc
T = 1; Fs =1/T; % Dat T=1
OmegaP = (2/T)*tan(wp/2);
OmegaS = (2/T)*tan(ws/2);
% Tinh toan bo loc tuong tu:
[cs, ds] = afd_chb1(OmegaP,OmegaS,Rp,As);
% Bien doi song tuyen:
[bl,al] = bilinear(cs,ds,Fs);
%
wph = 0.6*pi;
alpha = cos((wpl-wph)/2) / cos((wpl+wph)/2)
Nz = -[-alpha, 1];
Dz = [1, -alpha];
% Chuyen doi bo loc
[bh,ah] = zmapping(bl,al,Nz,Dz);

[db,mag,pha,grd,w] = freqz_m(bh,ah);
%plot
figure(37); clf;

%plot
subplot(2,2,1); plot(w/pi,mag);
axis([0,1,0,1.2]); grid
title('Amplitude Response');
xlabel('frequency in pi units'); ylabel('|Hr(w)|');
%
subplot(2,2,3); plot(w/pi,db);
axis([0,1,-30,10]); grid
```

```
title('Magnitude Response');
xlabel('frequency in pi units'); ylabel('Decibels');
%
subplot(2,2,2); plot(w/pi,pha/pi);
axis([0,1,-1,1]); grid
title('Phase Response');
xlabel('frequency in pi units'); ylabel('Angle(Hr(w))');
%
subplot(2,2,4); plot(w/pi,grd);
axis([0,1,-10,10]); grid
title('Group Delay');
xlabel('frequency in pi units'); ylabel('Samples');
```

***Vẽ phác hoạ đồ thị vào phần trống dưới đây:***



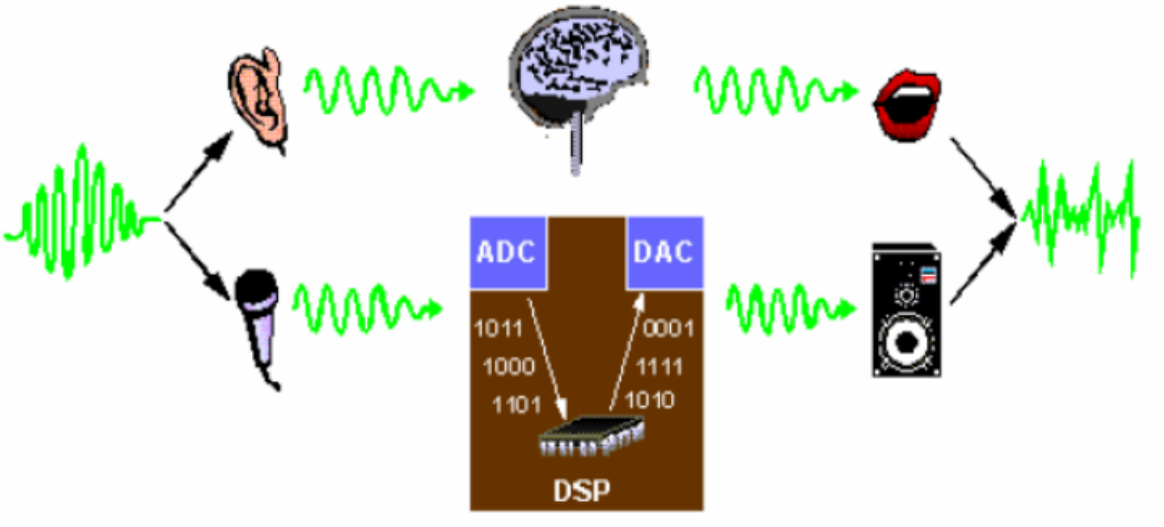
# BÀI 3. GIỚI THIỆU VỀ DIGITAL SIGNAL PROCESSOR

## 1. Mục đích:

Kết thúc bài thí nghiệm này, sinh viên có thể giải thích sự khác nhau giữa một bộ xử lý tín hiệu số(DSP) và một bộ xử lý mục đích chung. Xa hơn một bước, sinh viên có thể làm quen với quá trình thiết kế cho các chương trình cho DSP.

## 2. Cơ sở lý thuyết.

Bộ xử lý tín hiệu số(Digital Signal Processor - DSP) là một bộ phận xử lý mạnh và rất nhanh, nó có thể điều khiển quá trình phân tích tín hiệu trong thời gian thực. Bởi các phần tử khoá cho các mạch logic được thiết kế chuyên dụng cho các phép toán nhân và cộng nên thời gian tính toán trong các DSP nói chung thường nhanh hơn so với các bộ vi xử lý khác.



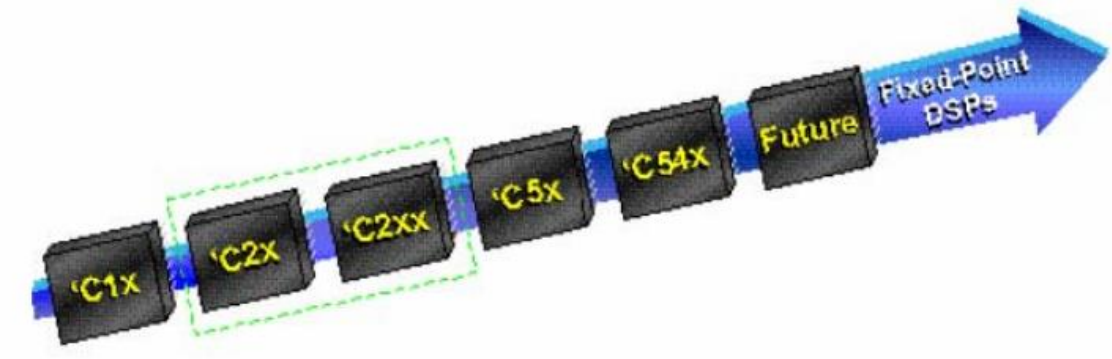

Các bộ xử lý tín hiệu số được đặc trưng bởi:

- Các cấu trúc chuyên môn hoá cho phép chúng thực hiện các lệnh mới một cách nhanh chóng và hiệu quả
- Các chỉ thị nhận nhanh

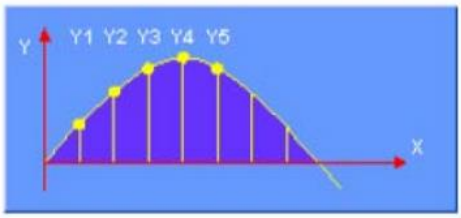
- Một số rút gọn các lệnh làm cho quá trình lập trình DSP đơn giản hơn

Các DSP đã làm cuộc cách mạng trong công nghệ điện tử viễn thông. DSP có thể coi như trái timtrong hàng loạt các thiết bị hiện đại như điện thoại di động, các thiết bị nhận dạng và tổng hợp tiếng nói, bộ chơi DVD (Digital Versatile), và các thiết bị an toàn mức cao. Không những vậy, rất nhiều ứng dụng ngày nay đã được tích hợp DSP như là trung tâm điều khiển của hệ thống bao gồm các bộ điều khiển đĩa cứng, các hệ thống treo xe ô tô, trong các mạng xử lý tín hiệu ảnh y tế, và các hệ thống radar.

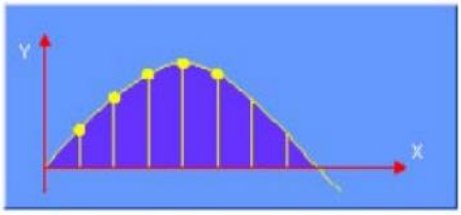
DSP bắt đầu xuất hiện vào cuối những năm1970 và vào đầu năm1980 với DSP1 của Bell Lab, 2920 của Inlel, uPD7720 của NEC. Vào năm 1982, Texas Instrument đã đưa ra TMS32010, thành viên đầu tiên của họ DSP dấu phẩy tĩnh 16 bit. DSP này có tốc độ tính toán là 8MIPS. Các bước nhảy vọt liên tiếp xuất hiện. Cụ thể là vào năm 1998, các DSP sử dụng xử lý song song đã đạt tới tốc độ tính toán 1600MIPS.

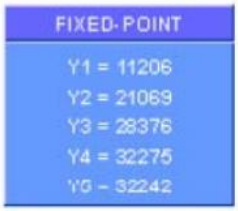


Trong hệ thống thí nghiệm Lab-Volt DIGITAL SIGNAI PROCESSOR, loại DSP được sử dụng là Texas Instrument TMS320C50. Đây là loại DSP thế hệ thứ ba với thiết kế bên trong dựa trên DSP thế hệ thứ nhất TMS320C10.







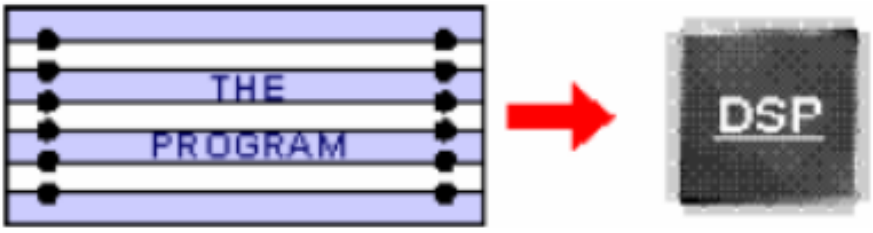




Cũng vào năm1982, các bộ xử lý dấu phẩy động đầu tiên đã được sản xuất bởi Hitachi. Khuôn dạng số này tăng đáng kể khoảng tính toán động của DSP. Hai năm sau NEC đã đưa ra các DSP 32 bit dấu phẩy động đầu tiên có tốc độ tính toán 6,6MIPS.

Nói chung, các tín hiệu của thế giới thực (ví dụ: âm thanh, radar) được xử lý tốt hơn bằng các DSP dấu phẩy động. Các tín hiệu được xây dựng (ví dụ như: viễn thông, ảnh và điều khiển) nói chung được xử lý tốt hơn bằng các DSP dấu phẩy tĩnh .

Trên thếgiới, xu thếphát triển các sản phẩm dựa trên DSP tăng nhanh vì:

- Chúng cho phép xử lý phức tạp hơn các mạng tương tự.
- Chúng cung cấp tính năng xử lý tín hiệu lặp đi lặp lại.
- Mã nguồn có thể dễ dàng được sửa đổi và việc cập nhật. Nói một cách khác, thay đổi thiết kếcủa nó là mềm dẻo hơn.
- Chúng thường được cho giá thành phát triển thấp hơn các thiết kế tương tự với các bậc tính năng tương đương.




`Một hệ thống muốn vận hành cần phải thông qua sự chỉ thị từ một phần mềm được lập trình từ trước. Phần mềm bao gồm một tập các chỉ dẫn, hay còn gọi là các lệnh, để bảo cho hệ thống biết sẽ làm các công việc gì một cách tuần tự và hệ thống cần thao tác thế nào một khi có một điều kiện đã được dự đoán trước xảy ra.. Chương trình này được lưu trữ như mã máy bên trong DSP.

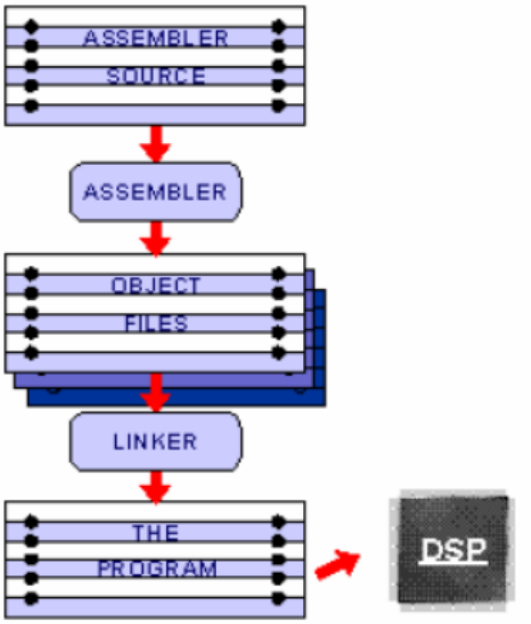
**Hỏi:** Lựa chọn nào trong các lựa chọn dưới đây là một lệnh nằm trong chương trình?

a. ADD #214, 4

b. F9E7h

c. 1011,1110 0001 0110

d. Tất cả các lựa chọn trên



Xây dựng một chương trình DSP mà đơn thuần từ mã máy là không khả thi. Vì lý do này, ngôn ngữ assembler (hợp ngữ) được phát triển để viết chương trình cho DSP. Đây là ngôn ngữ lập trình mà các chỉ thị của nó ở dạng gợi nhớ là biểu tượng và thường tương ứng một – một với các chỉ thị máy.

Bộ dịch (assembler) và bộ liên kết (linker) được sử dụng để dịch chương trình được viết bằng hợp ngữ thành các mã máy của DSP. Assembler dịch tệp chương trình thành tệp đích, các tệp này sau đó được liên kết với nhau (link) để tạo ra tệp mã máy vận hành bên trong DSP.

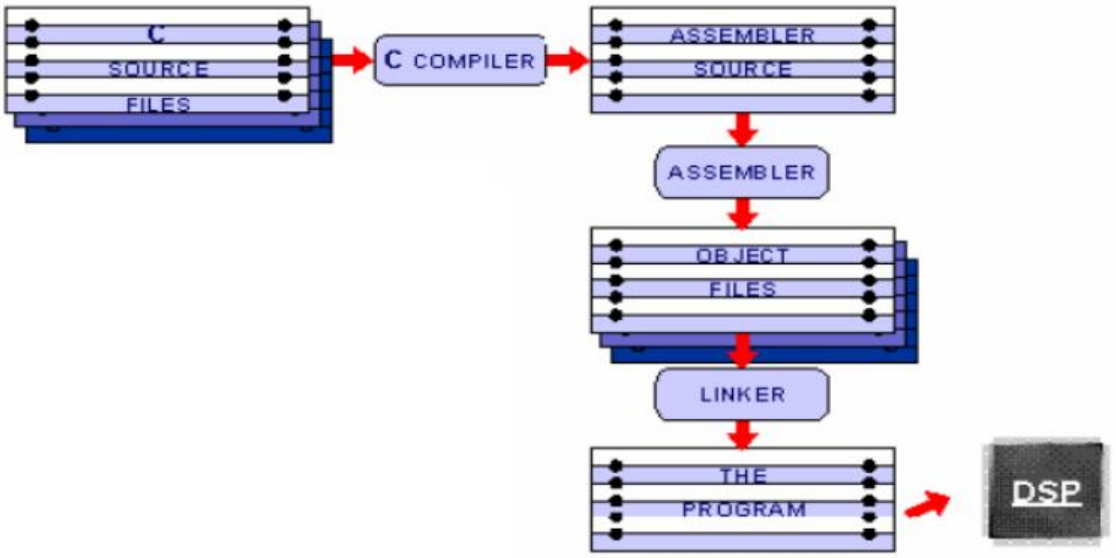
**Hỏi:** Sự lựa chọn nào trong các câu lệnh dưới đây được viết bằng hợp ngữ?

a. IF (i.NE.27) THEN(omega=2*sin(x))

b. 982Eh

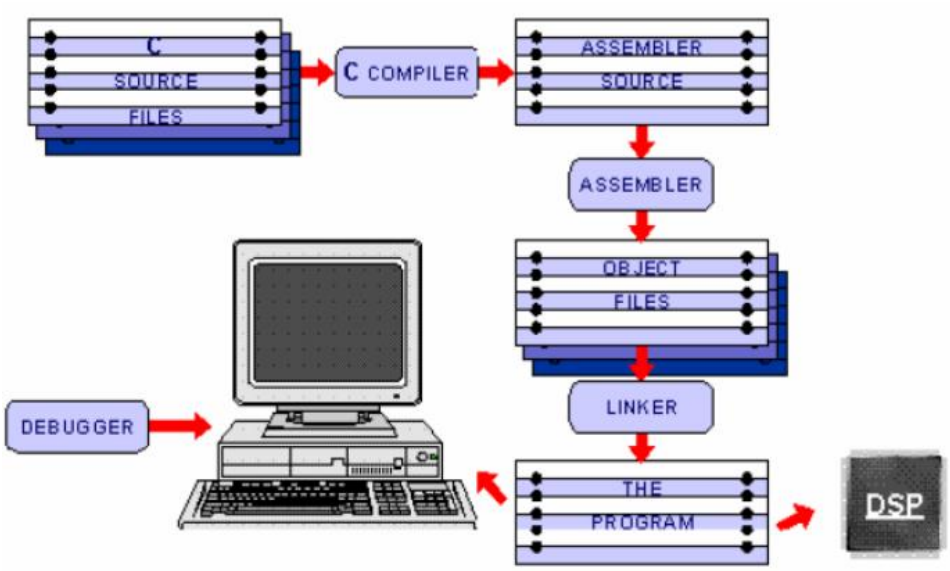
c. 1011 1110 0001 0110

d. DMOV *, AR1



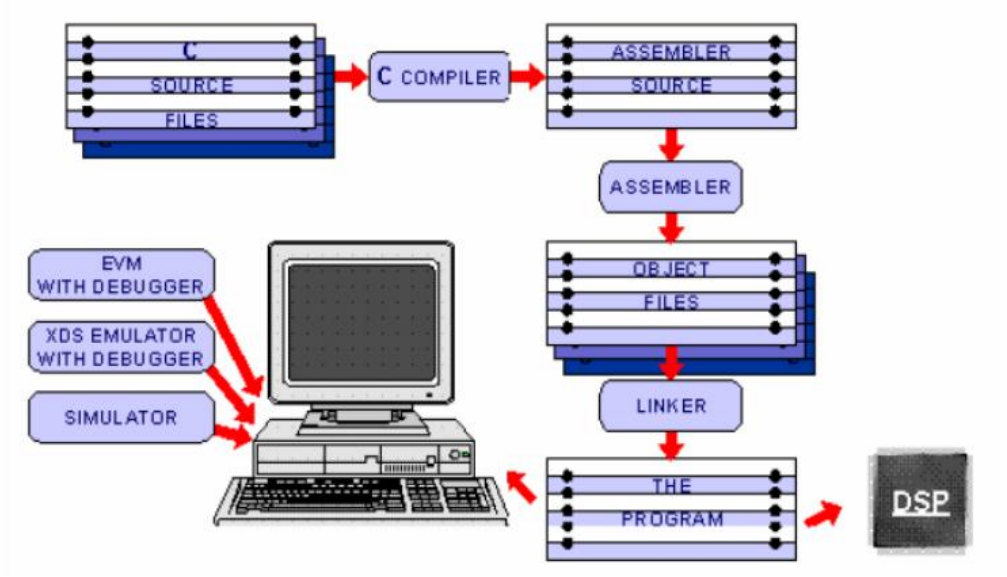

Ngôn ngữ C là ngôn ngữ bậc cao được sử dụng ngày càng nhiều để lập trình các DSP phức tạp hoặc thực thi các thuật toán có độ phức tạp cao. Lập trình bằng C đơn giản hoá thiết kế của các ứng dụng DSP vì người lập trình không còn bị giới hạn bởi tập chỉ thị nhỏ của các ngôn ngữ bậc thấp (như hợp ngữ).

Bộ biên dịch (compiler) C được sử dụng để dịch các mã nguồn C thành các mã hợp ngữ DSP thích hợp.




Phần cuối của lập trình bao gồm việc kiểm tra lỗi chương trình và làm thay đổi cho đến khi thực hiện tốt chức năng mong muốn. Quá trình cuối cùng trong chuỗi các quá trình phát triển một phần mềm thường được gọi là gỡ rối (debugging). Chương trình giúp cho việc gỡ rối phần mềm được gọi là bộ gỡ rối (debugger).

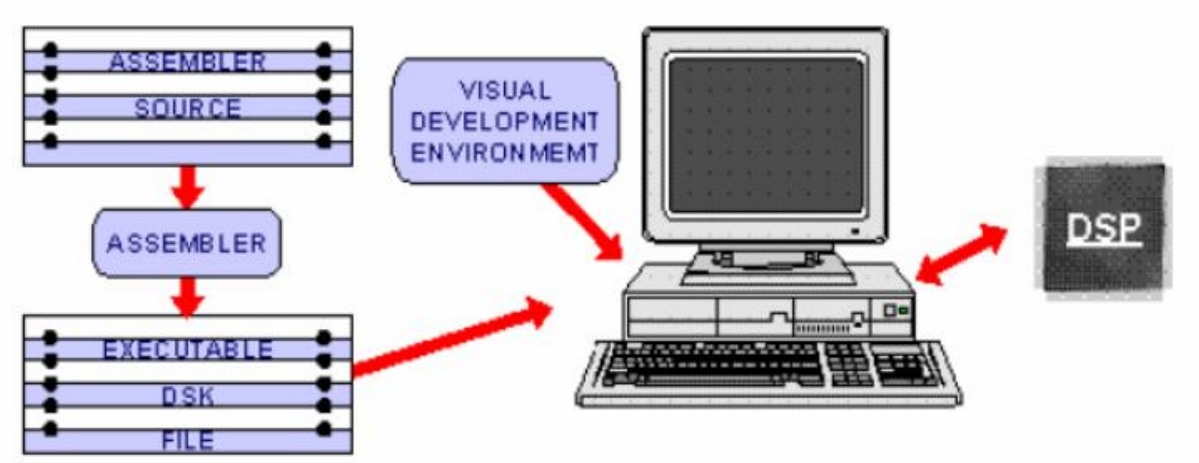
Một bộ gỡ rối cho phép người lập chương trình khả năng phân tích vấn đề kết hợp với các chương trình DSP của họ. Điều này được thực hiện trước khi gỡ rối được sử dụng với DSP mà ta làm thí nghiệm.C5x Visual Development Evironment (C5x VDE) là bộ gỡ rối được sử dụng với DSP mà chúng ta làm thí nghiệm.




Những người phát triển hệ thống DSP hiếm khi gỡ rối một DSP mà không sử dụng một bộ gỡ rối hay debugger. Vì vậy, họ thường sử dụng EVMs, emulators và simulators để trợ giúp cho việc này.

Bộ DSP được sử dụng với bộ mạch là một bộ phận của module TM320C5x DSK (Digital Signal Processing Kit). Khi sử dụng EVMs, emulators và simulators, người phát triển có thể thay đổi trong quá trình phát triển mô hình của DSP đang được thí nghiệm.

Một khi đã hoạt động được, thử nghiệm cuối cùng của chương trình này được cài đặt trên hệ thống DSP.



Các chương trình được bao gồm và sử dụng trong Digital Signal Processor được viết bằng hợp ngữ. Hợp ngữ được sử dụng như một đặc trưng của TM320C5x EVMs, nó đã cộng thêm các chỉ thị trong nó, và được gọi là các chỉ thị DSK.

## 3. Yêu cầu thiết bị

Để hoàn thành được các bài tập sau đây, ta cần:

- FACET baseunit.
- Bọ mạch DIGITAL SIGNAL PROCESSOR.
- Chương trình C5x VDE.
- Các tệp chương trình (dsk) và hợp ngữ(asm) 1_1, Exl_2
- Máy hiện sóng
- Đồng hồ đo điện đa chức năng

*********

# BÀI 4. LÀM QUEN VỚI BỘTHÍ NGHIỆM LABVOLT - DSP

## 1. Mục đích

Kết thúc bài này, sinh viên được làm quen với vị trí và chức năng của mỗi linh kiện khác nhau trong hệ thống DSP

## 2. Thảo luận

Bo mạch có hai vùng chức năng: vùng chứa các phụ kiện của bo mạch và vùng chứa DSP và ngoại vi của nó.

***Vùng chứa các phụ kiện của bo mạch bao gồm:***

- DOWER SUPPLY với AUXILIARY POWER INPUT
- DC SOURCE.

• MICROPHONE PRE-AMPLIFIEF
• AUDIO AMPLIFIER

Chức năng:

• Khối mạch POWER SUPPLY cung cấp một nguồn DC đã được chỉnh lưu và lọc cho toàn bộ bo mạch. Bo mạch có thể được vận hành theo hai cách khác nhau : hoặc điện áp vào của Power Supply có thể được nhận từ Lab-BoIl FACET base Unit hoặc có thể được nhận từ các kết nối ± 15V ngoài được tìm thấy trên khối AUXILIARY POWER INPUT.
• Khối DC SOURCE cung cấp một điện áp DC thay đổi và phụ thuộc vào vị trí của chiết áp,giữa -3,5V de và + 3,5Vdc. Khối DC SOURCE có thể được dùng nguồn của một tín hiệu tham chiếu đầu vào cho chương trình chạy trên DSP.
• Khối MICROPIIONE PRE-AMPLIFIER được sử dụng để điều chỉnh một tín hiệu micro thành một mức thích hợp với đầu vào của DSP. Chiết áp GAIN thay đổi mức ra giữa một giá trị thấp và một giá trị cao.
• Để có thể nghe thấy tín hiệu từ ANALOG OUTPUT, được định vị trên khối CODEC, khối AUDIO AMPLIER được sử dụng.

***Vùng chức năng thứ hai của bo mạch là DSP và các ngoại vi của nó bao gồm:***

• DSP
• CODEC
• I/O INTERFACE
• INTERRUPTS
• AUXILIARY I/O
• SERIAL PORT.

DSP được coi như là trái tim của hệ thống xử lý tín hiệu số.

• Khối DSP chứa một vi mạch DSP TM320C50 trong một chíp 132 chân dán trên bề mặt (surface mount). Nó có thể đạt tới tốc độ thực hiện 50MIPS. Có

nhiều lại DSP chúng có thể thay đổi về các tốc độ chu trình. Tuy nhiên, tốc độ được giới hạn bởi các ràng buộc của hệ thống bên trong vi mạch. DSP có thể sử dụng một bộ tạo dao động bên trong để thiết lập đồng hồ hoặc cũng có thể sử dụng bộ tạo dao động ngoài. DSP được dùng trên bo mạch thí nghiệm được đặt cấu hình để sử dụng bộ tạo dao động ngoài.

• Khối OSCILATOR được đặt trên bo mạch cung cấp cho nó một tín hiệu tham chiếu 40 MHz. DSP chia tín hiệu này để tạo ra tín hiệu bên trong 20Mhz (tần số tín hiệu chủ) mà nó sử dụng để tính toán thời gian các chu trình chỉ thị của nó.

• Khối CODEC thường được cấu thành bởi các linh kiện sau:

- một đầu vào GAIN lập trình được
- một ANTI-ALISING FILTER (bộ lọc chống trùm phổ)
- một bộ biến đổi tương tự - số
- một bộ biến đổi số - tương tự
- một POST-GILER (bộ lọc sau)

• Khối I/O INTERFACE là một phương tiện để hiển thị và nạp và thông tin chương trình. Chuyển mạch DIP8 có chức năng đưa 8 bit vào cấu hình DSP. Phụ thuộc vào chương trình đang được sử dụng, thông tin có thể được xử lý theo nhiều cách khác nhau. Các bộ hiển thị LED 7 thanh được sử dụng để đưa ra thông tin chương trình cho người sử dụng DSP. Như hầu hết các bộ vi xử lý, các DSP đều có khả năng điều khiển ngắt. Hai nút có thể được sử dụng như các thiết bị vào của người sử dụng cho một chương trình. Khi một trong các nút nhấn được nhấn thì một ngắt được sinh ra bên trong DSP và mã chương trình kết hợp với nó được thực hiện.

• Vùng AUXILARY I/0 đã được cộng thêm vào cho mục đích giám sát tín hiệu và để làm nguyên mẫu cho các bài tập DSP thêm vào được thực hiện trên bo mạch. Các đầu của khối AUXILARY I/O có thể được sử dụng để giao

tiếp DSP với một mạch ngoài. Mạch ngoài này có thể được cấp nguồn bởi đầu 10 chân đặt trên khối AUXILARY I/O. Vùng AUXLIIARY I/O có ba cổng:

- Các điểm kết nối ± 5Vdc và ± 5Vdc có sẵn để sử dụng trên đầu phải có 10 chân, chúng có thể được sử dụng để cấp nguồn cho một mạch ngoài. Các bộ cung cấp của bo mạch có cùng điểm đặt.
- Đầu trái của 8 chân LSB (được đánh nhãn từ D0 đến D7) của bus dữ liệu của DSP ngoài, và bao gồm 4 đường địa chỉ được tiền mã hoá (được đánh nhãn từ PA0# đến PA3#).
- Đầu giữa có các phần vào/ra (I/O) bao gồm:
    - chọn dữliệu(DS#), chương trình (PS#), khoảng vào/ra (IS#)
    - đầu ra bộ định thời
    - chọn đầu (RD#) và cho ghi (WE#) cho các thiết bị ngoài
    - chọn đọc/ghi (R/W#) cho các truy nhập ngoài.
    - tín hiệu báo cho biết đã nhận được ngắt (IACK#)
    - đầu vào ngắt ngoài (INT4#)
    - chọn hướng (DIR) và chọn chíp (CS#) để điều khiển việc truyền dữ liệu ngoài.

DSP trên bo mạch được lập trình để thành vai trò server đối với máy tính trong vai trò client. Để bộ DSP hoạt động, bo mạch SERIAL, PORT phải được nối với một trong các cổng nối tiếp của máy tính của bạn.

**Chú ý:** Nếu máy tính chủ không có một kết nối tiếp thứ hai thì vào thời điểm thích hợp trong tiến trình thực hiện bài tập sinh viên có thể tháo kết nối tiếp của Base Unit và dùng nó để nối bo mạch SERIAL PORT với máy tính C5x VDE (C5x Visual Development Environment) quản lý việc bắt tay giữa bo

mạch và máy tính. Nó điều khiển tất cả các đầu vào và đầu ra từ bộ nhớ của DSP cổng nối tiếp. Một khi kết nối liên lạc giữa máy tính của bạn và bo DSP được thiết lập, C5x VDE có thể được sử dụng để nạp một chương trình vào DSP.

## 3. Tiến trình thí nghiệm

**Giới thiệu bo mạch:** Trong phần này, bạn sẽ làm quen với một số các linh kiện và khối mạch trên bo mạch DIGIAL SIGNAL PROCESSOR.

1. Định vị trên bo mạch DIGITAL SIGNAL PROCESSOR tất cả các thiết bị đầu cuối chung. Dùng một điện trở kế để kiểm tra các thiết bị đầu cuối được nối với nhau hay chưa.
2. Bật nguồn cung cấp cho bo mạch DIGITAL SIGNAL PROCESSOR.
3. Dùng một volt kế để kiểm tra điện áp một chiều bằng cách thay đổi chiết áp của DC SOURCE từ giá trị nhỏ nhất cho tới giá trị lớn nhất của nó. Đo điện áp DC tại đầu ra của DC source

| |
|---|
| **Hỏi:** Điện áp DC nhỏnhất (VDC min) và điện áp DC lớn nhất (VDC max) đưa ra từ<br>DC source?<br>VDC min = ……V<br>VDC max = ……V |

4. Thực hiện các kết nối với DIGITAL SIGNAL PROCESSOR

**Chú ý:** Nếu chất lượng audio từ loa không tốt, có thể dùng tai nghe kèm theo bo mạch. Nối tai nghe vào đầu cắm tai nghe được đặt trên khối mạch AUDIO AMPLIFIER.

5. Nói vào micro, xem xét sự thay đổi của âm thanh phát ra trong khi cùng thực hiện thay đổi chiết áp của MICROPHONE PRE-AMPLIFIER và của AUDIO AMPLIFIER
6. Tháo toàn bộ các kết nối hiện có trên bo mạch.

**Làm quen với bo mạch dùng một chương trình DSP:**Trong mục này, C5x VDE sẽ được dùng để nạp và chạy một chương trình bên trong DSP

**Chú ý:** Trước khi sửdụng C5x VDE, hãy chắc chắn rằng nguồn của bo mạch được bật và kết nối nối tiếp là hiện có giữa máy tính và khối mạch DIGITAL SIGNAL PROCESSOR được đánh nhãn SERIAL PORT.

7. Mở chương trình C5x VDE:

8. Dùng lệnh Load Program trong menu File để nạp chương trình ex1_1.dsk vào DSP.

**Hỏi:** Hai cửa sổnào đang được mởtrong C5x VDE?

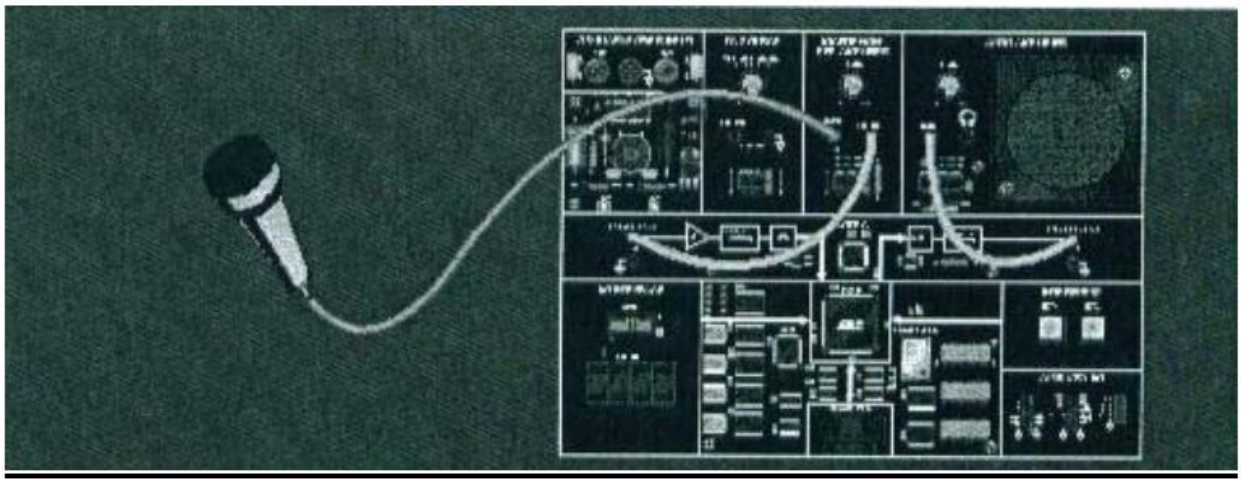
a. C5x Registers và Peripheral Registers.

b. Dis-Assembly và Periphearal Registers.

c. C5x Registers và Dis-Assembly.

d. Peripheral Registers và File Selection

9. Kết nối bo mạch như hình vẽ. Điều này cho phép chương trình ex1_1.dsk vận hành đúng đắn.



**Chú ý:** Dùng tai nghe nếu cần thiết.

10. Thực hiện lện RUN trên thanh công cụ của C5x VDE.

11. Quan sát những gì đọc ra được hiển thị bên trong khối mạch I/O INTERFACE.

Điều chỉnh chuyển mạch DIP (tất cảcác bit đều ởvịtrí 0)sao chohiển thị đọc được là **0000**.

12. Nhấn nút thứ nhất INT# trên bo mạch INTERRUPTS để chuyển tới DSP các giá trị được nhập vào thông qua chuyển mạch DIP.

13. Dùng micro, cho một tín hiệu (giọng nói) vào DSP

**Chú ý:** Điều chỉnh các chiết áp GAIN của MICROPHONE PRE-AMPLIFIER và của AUDIO AMPLIFIER để cải thiện âm thanh đầu ra.

14. Lưu ý rằng trong khi đang nói vào micro, các chấmtrên màn hình của khối mạch I/O INTERFACE bật sáng.

15. Điều chỉnh chuyển mạch DIP sao cho màn hình I/O INTERFACE đọc được là **0015**.

16. Truyền giá trị của chuyển mạch DIP vào DSP bằng cách nhấn nút nhấn INT#.

17. Quan sát kết quả của sự thay đổi của xử lý tín hiệu trong âm thanh của giọng nói.

18. Lặp lại các bước từ 15 đến 17 cho mỗi một giá trị được hiển thị trên I/O INTERFACE sau đây: **0031**, **0063**, **0127**, **0255**

Nhớ nhấn nút INT # sau khi đặt chuyển mạch DIP tới một giá trịmới.

**Hỏi:** Sự lựa chọn nào sau đây là mô tả đúng đắn nhất về chương trình ex1_1.dsk được nạp vào DSP?

a. Đây là một bộ ghi tiếng nói

b. Đây là hệ điều hành Base Unit

c. Đây là một máy phát chức năng

d. Đây là một máy phát tiếng vọng.

**Hỏi:** Con số được hiển thị trên I/O INTERFACE tỉ lệ với cái gì?

a. Thời gian trễ(theo ms) giữa các tiếng vọng liên tiếp

b. Số các tiếng vọng được tạo ra

c. Thời gian cần dùng (theo ms) để sinh ra các tiếng vọng cho một âm thanh

d. Số các mẫu phải lấy trên tín hiệu ra trong một giây

19. Thực hiện lệnh Halt trên thanh công cụ của C5x VDE. Đóng C5x VDE.

## 4. Kết luận

• DIGITAL SIGNAL PROCESSOR có hai vùng: vùng các phụ kiện của bo mạch và vùng DSP với các ngoại vi.

• Bo mạch được chia thành các khối mạch riêng rẽ.

• Trước khi một chương trình DSP có thể được nạp hoặc sử dụng, nguồn cung cấp của DIGITAL SIGNAL PROCESSOR phải được bật lên và kết nối nối tiếp giữa khối mạch SERIAL PORT và máy tính phải được thực hiện

• Các khối mạch CODEC, I/O INTERFACE, INTERRUPTvà AUXILIARY I/O có thể chỉ được áp dụng bởi người sử dụng nếu chương trình nạp vào DSP đòi hỏi việc sử dụng chúng.

## 5. Câu hỏi ôn tập

Dưới đây là các câu hỏi cho Bài 4. Sinh viên đọc kỹ câu hỏi, sau đó tích vào ô tương ứng với câu trả lời được cho là đúng nhất:

**Câu 1:** Trước khi bo mạch DIGITAL SIGNAL PROCESSOR sẵn sàng để sử dụng, có một số bước bắt buộc cần phải theo. Mệnh đề nào sau đây là bước cần thiết phải thực hiện trước khi sử dụng bo mạch ?

a. Chắc chắn rằng các chuyển mạch của I/O INTERFACE đều ở vị trí 0

b. Chắc chắn rằng kết nối nối tiếp là hiện có giữa máy tính chủvà khối mạch DIGITAL, SIGNAL PROCESSOR được đánh nhãn SERIAL PORT.

c. Chắc chắn rằng nguồn cung cấp của bo mạch được bật

d. Các mệnh đề b và c.

**Câu 2:** Khoảng điện áp DC mà chiết áp cho nguồn DC điều chỉnh được là bao nhiêu?

a. –3,3V đến +3,6V

b. –3,0V đến + 3,0V

c. –3,5V đến + 3,5V

d. Không có mệnh đề nào trong các mệnh đề trên là đúng.

**Câu 3:** Chân nào trong số các chân sau đây được đặt trên đầu giữa của bo mạch AUXILIARY I/O ?

a. 4 đường địa chỉ tiền mã hoá (được đánh nhãn từ PA0# đến PA3#)

b. TOUT, IACK #, INT4#, và RD#

c. DS#, D0, D1, và D2

d. CS#, INT4#, DS#, và PA1#

**Câu 4:** DSP TMS320C50 trên bo mạch DIGITAL SIGNAL PROCESSOR sử dụng đồng hồ hệ thống có tần số là bao nhiêu (nhắc lại rằng đây là đồng hồ đặt tốc độ tính toán cho DSP)?

a. DSP dùng bộ tạo dao động bên trong 20MHZ

b. DSP dùng bộ tạo dao động bên ngoài 40MHZ

c. Thông qua kết nối nối tiếp, DSP dùng bộ tạo dao động bên trong 33.3MHz CODEC

d. Thông qua kết nối bo mạch SERIAL PORT, DSP dùng bộ dao động trong của máy tính chủ.

**Câu 5:** Linh kiện nào trong các linh kiện sau đây thường được tìm thấy trong CODEC

a. Một bộ lọc chống trùm phổ

b. Một bộ biến đổi tương tự – số

c. Một bộ biến đổi số – tương tự

d. Tất cả các bộ nói trên.