



Universitatea POLITEHNICA din București
Facultatea de electronică, Telecomunicații și Tehnologia Informației



TEMA 1

PARTEA 1.A

PROCESOARE DE SEMNAL IN COMUNICATII

Nume student: Stanescu Vlad-Constantin

Grupa: 444C

1. Cerinta temei

I.A. Realizați un proiect C pentru StarCore140 care să implementeze o **funcție** (apelată din main) cu operația descrisă în tabelul de mai jos (fiecare student are o temă). Se va porni de la proiectul dat ca exemplu pe site-ul cursului. Programul și funcția C vor folosi tipurile de date asociate numerelor fracționare și funcțiile intrinseci specifice DSP StarCore140.

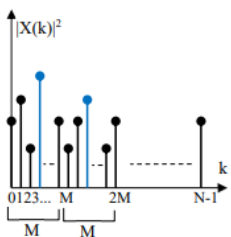
De asemenea se va realiza un program de test Matlab care va genera un număr suficient de valori pentru a se testa programul C pentru SC140.

Valorile de test generate în Matlab vor fi scrise în fișier/fișiere și vor fi citite de funcția main din proiectul C. Apoi se va apela funcția implementată și rezultatele vor fi scrise (tot din main) în fișiere. În Matlab se compară apoi rezultatele obținute, evaluându-se inclusiv erorile datorate formatelor de reprezentare.

Tema I.A. se va salva într-un director (folder) cu numele **44gs_IA_Nume_Prenume** (unde 'gs' e grupa și seria).

Tema I.A. va conține:

- un fișier Word cu: descrierea programului, detalii de implementare, rezultatele testării și explicații suplimentare.
- un director cu proiectul StarCore integral (cu programele C și toate subfolderele pe care le are și proiectul dat ca model pe site, *Exemplu proiect tema 1*, inclusiv un subdirector cu programele Matlab de test).

9	<pre>void maxim_spectr(Word16 *X, Word16 *out, unsigned short N, unsigned short M, unsigned short P);</pre>	<p>Considerând N componente complexe ale unui vector X (de exemplu componentele spectrale calculate cu TFD de lungime N) să se determine <u>valorile maxime</u> ale modulului la pătrat ale componentelor <u>din fiecare set</u> (subbandă) de M componente.</p> <p>Se consideră N multiplu de M și se va declara (pe lângă valorile M și N) și constanta $P=N/M$ pentru a evita operația de împărțire în program.</p> <p>N, M și P trebuie alese astfel încât să fie multiplu de 4 (de ex. $N=1024$, $M=16$ și rezultă $P=64$).</p>  <p>Word16 *X → pointer la vectorul cu componente complexe. Părțile reale și imaginare ale elementelor vectorului sunt stocate astfel: {X_re(0), X_im(0), X_re(1), X_im(1), ..., X_re(N-1), X_im(N-1)}. În total vectorul X are lungime 2N.</p> <p>Word16 *out → pointer la vectorul de ieșire (conține cele P valori maxime).</p>
---	---	--

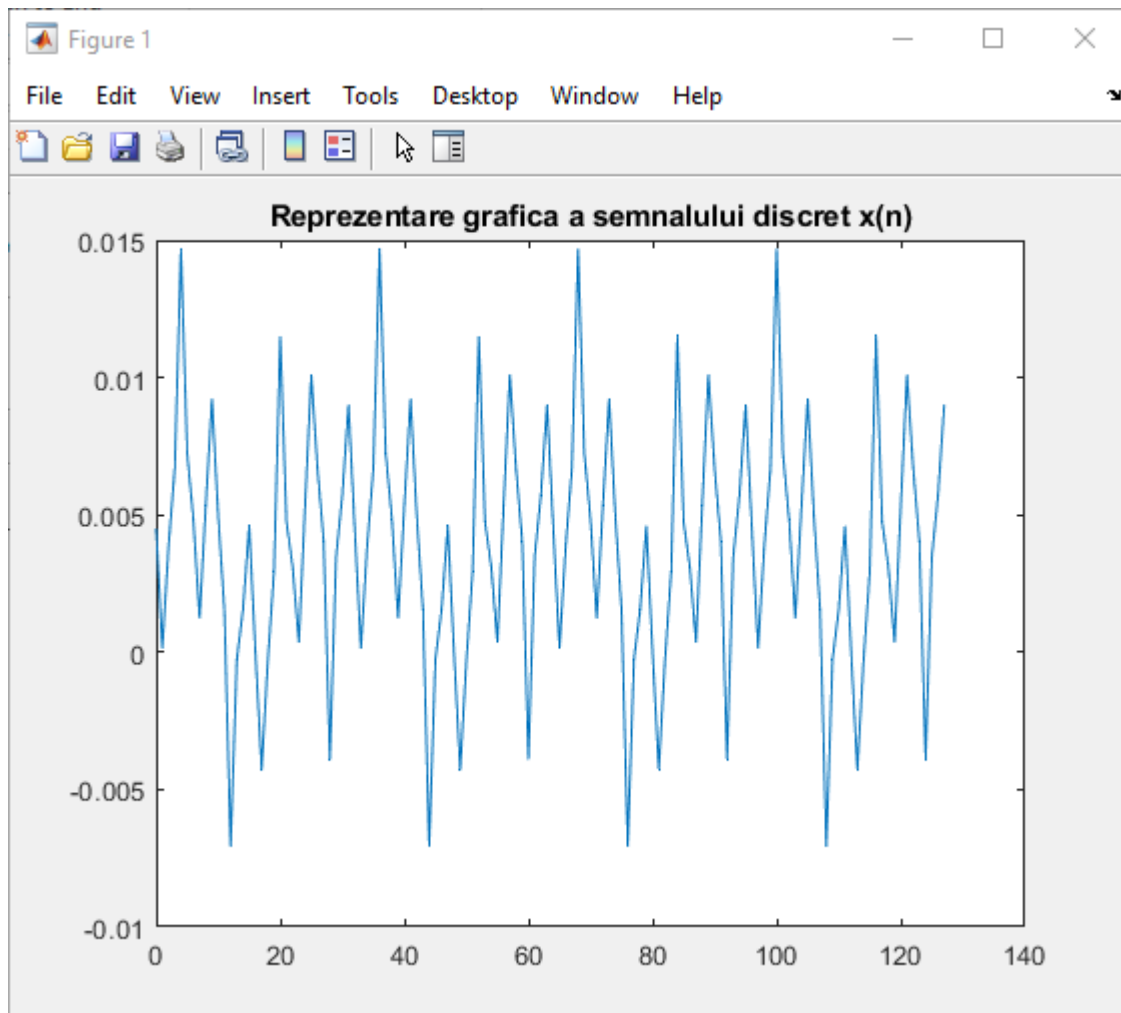
Pentru realizarea temei am ales valorile **N=128** (nr. componente complexe ale vectorului X), **M=16** (nr. componente din fiecare subbandă), **P=8** (nr. subbenzi) .

2. Generarea semnalului de intrare si prelucrarea acestuia

Pentru prelucrarea vectorului in Code Warrior am tinut cont de formatul de date Word16 si am ales un semnal de intrare astfel incat valorile rezultate (valorile maxime ale modulului la patrat) sa se regaseasca in intervalul [0, 1]. Am ales un semnal de intrare aleatoriu, alcatuit dintr-un sinus si un cosinus, avand amplitudini mai mici sau egale cu 0.01 pentru a nu afecta prelucrarea vectorului de maxime in Code Warrior.

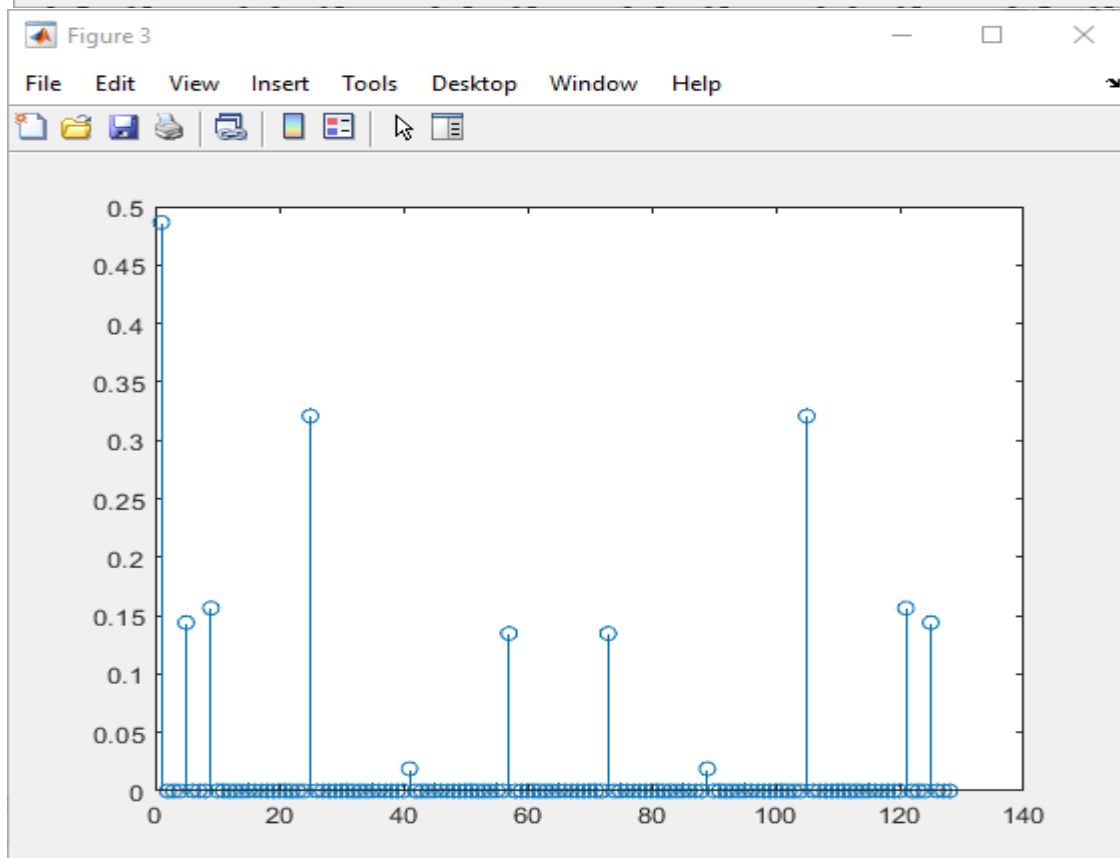
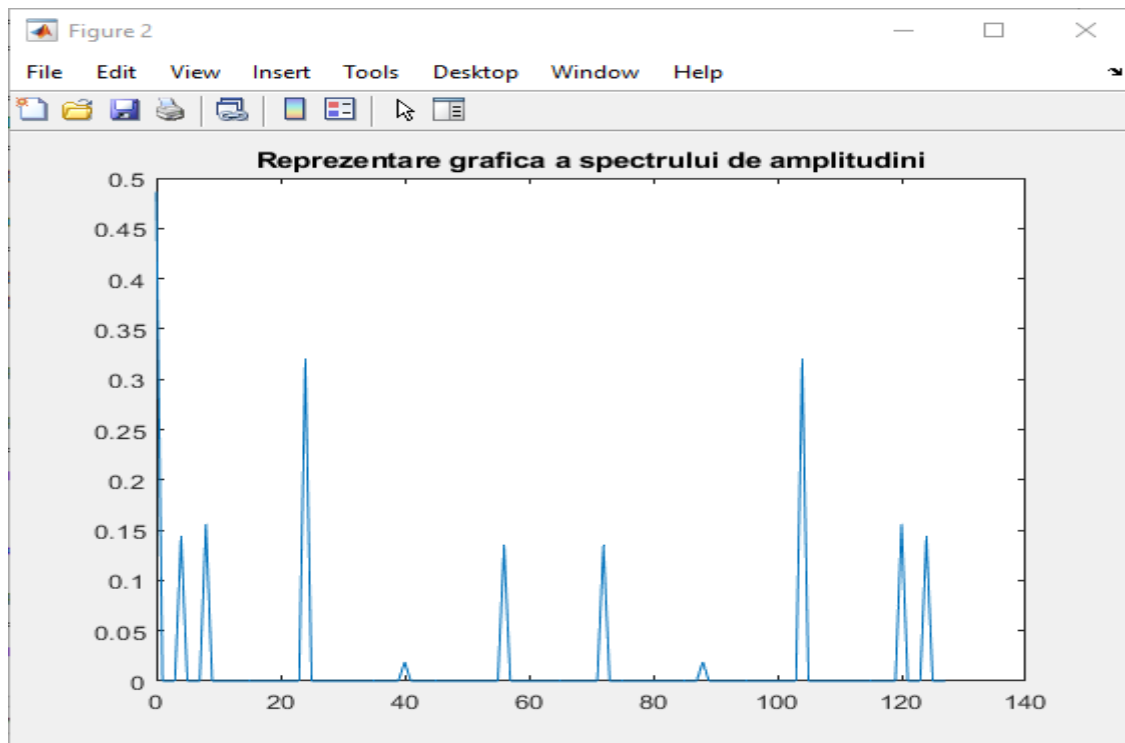
```
% Definire parametrii
% Vom alege M, N, P astfel incat acestea sa fie multiplii de 4:
N = 128; %nr. componente complexe ale vectorului X
M = 16; %nr. componente din fiecare subbanda
P = N/M; %nr. subbenzi: P = 128/16 = 8

n = 0 : N-1; % vectorul pentru indecsii de esantioane din vectorul de intrare, valori intre [0,255]
% Definire semnal discret: am presupus un semnal de intrare format dintr-o combinatie de sinus si cosinus
x = 0.0027*sin((2*pi*n)/M).^7 + 0.0031*(sin(pi*n/M)).^2 + 0.0045*(cos(pi*n/(2*M))).^2 + 0.0066*sin((26*pi*n)/M).^5;
figure(1), plot(n, x), title('Reprezentare grafica a semnalului discret x(n)');
```



- **Determinare FFT a semnalului discret:**

```
% Calculam transformata Fourier discreta a semnalului x(n)
X = fft(x, N);
figure(2), plot(n, abs(X)), title('Reprezentare grafica a spectrului de amplitudini');
figure(3), stem(abs(X));
```



Word16 *X → pointer la vectorul cu componente complexe. Părțile reale și imaginare ale elementelor vectorului sunt stocate astfel: {X_re(0), X_im(0), X_re(1), X_im(1), ..., X_re(N-1), X_im(N-1)}. În total vectorul X are lungime 2N.

Word16 *out → pointer la vectorul de ieșire (conține cele P valori maxime).

Având în vedere că în cerință se specifică că părțile reale și imaginare ale elementelor vectorului sunt stocate în forma de mai sus și că vectorul X are lungime 2N, am creat un vector Xc ce conține 256 de elemente. Am stocat partea reală a fiecărei componente din vectorul X la indicii impari ai vectorului Xc și partea imaginară la indicii pari ai vectorului Xc.

```
Xc = zeros(1,(2*N)); % vectorul cu componente complexe ce conține 2*N = 2*128 = 256 elemente, componentele fiind initializate cu 0

% Pentru partile reale vom modifica doar indicii impari ai vectorului Xc
k = 1;
for i = 1 : 2 : 2*N - 1
    Xc(i) = real(X(k));
    k = k + 1; % cu ajutorul acestui contor parcurgem vectorul X
end

% Pentru partile imaginare vom modifica doar indicii pari ai vectorului Xc
k = 1;
for j = 2 : 2 : 2*N
    Xc(j) = imag(X(k));
    k = k + 1;
end
```

- **Calculul valorilor maxime ale modulului la patrat ale componentelor din fiecare set de M = 16 componente**

```
%% Calculul valorilor maxime ale modulului la patrat ale componentelor din fiecare set de M = 16 componente

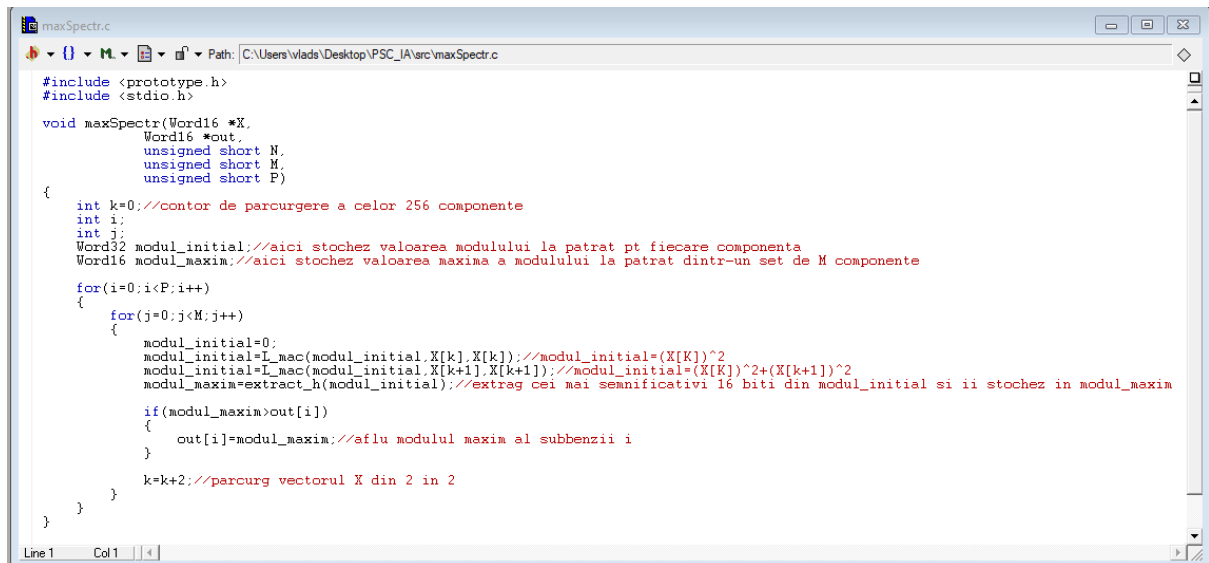
% Cream un vector pentru a stoca maximele modulelor la patrat:
maxime = zeros(1,P); % avem P = 8 maxime in total
p = 1; % contor pentru indicii vectorului maxime
for i = 1 : P
    module = zeros(1,M); % vector in care stocam valorile modulelor la patrat pentru fiecare sectiune de M componente spectrale
    k = 1; % contor pentru indicii vectorului module
    for j = (M*(i-1) + 1) : M*i % parcurgem fiecare subbanda
        module(k) = (real(X(j))^2) + (imag(X(j))^2);
        k = k + 1;
    end
    maxime(p) = max(abs(module)); % aflam maximul dintre cele M module la patrat si il introducem in vectorul maxime
    p = p + 1;
end
```

Am creat un vector „maxime” în care vom stoca toate valorile maxime din fiecare subbandă, apoi am creat 2 bucle FOR: prima este pentru a parcurge vectorul de maxime ce are P = 8 componente, iar cea de-a doua este pentru a parcurge fiecare set de M = 16 valori. După fiecare parcurgere a câte M componente spectrale, vom afla și maximul dintre toate modulele la patrat, și îl vom stoca în vectorul maxime.

- **Urmează scrierea vectorului Xc în fișierul x.dat, folosit pentru a prelua input-ul în Code Warrior:**

```
fid=fopen('..\x.dat','w','b'); % deschiderea fișierului x.dat în care va fi stocat vectorul de intrare
fwrite(fid,Xc.*2^15,'int16'); % scrierea în fișier și convertirea valorilor în format CW
fclose(fid); % închiderea fișierului
```

3. Prelucrarea componentelor complexe ale vectorului de intrare in CW



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

void maxSpectr(Word16 *X,
               Word16 *out,
               unsigned short N,
               unsigned short M,
               unsigned short P)
{
    int k=0; //contor de parcurgere a celor 256 componente
    int i;
    int j;
    Word32 modul_initial; //aici stochez valoarea modulului la patrat pt fiecare componenta
    Word16 modul_maxim; //aici stochez valoarea maxima a modulului la patrat dintr-un set de M componente

    for(i=0; i<P; i++)
    {
        for(j=0; j<M; j++)
        {
            modul_initial=0;
            modul_initial=L_mac(modul_initial, X[k], X[k]); //modul_initial=(X[k])^2
            modul_initial=L_mac(modul_initial, X[k+1], X[k+1]); //modul_initial=(X[k])^2+(X[k+1])^2
            modul_maxim=extract_h(modul_initial); //extrag cei mai semnificativi 16 biti din modul_initial si ii stochez in modul_maxim

            if(modul_maxim>out[i])
            {
                out[i]=modul_maxim; //aflu modulul maxim al subbenzii i
            }

            k=k+2; //parcurs vectorul X din 2 in 2
        }
    }
}
```

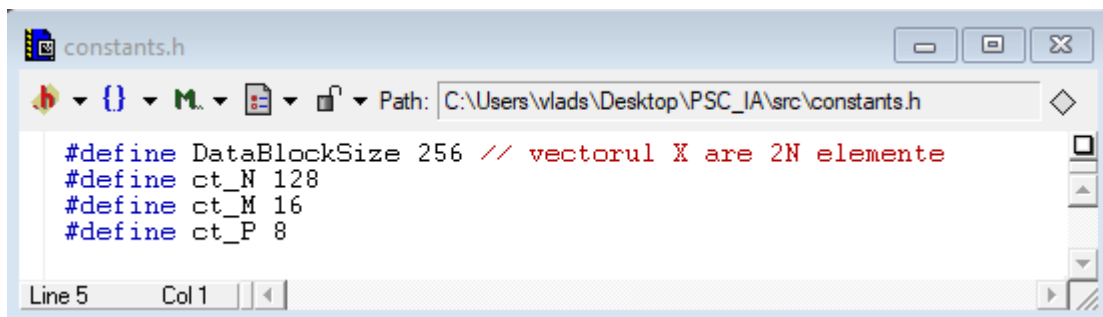
In functia **maxSpectr** am creat un contor k pentru a parcurge vectorul X, variabila de tip Word32 modul_initial pentru a stoca valoarea modulului la patrat pentru fiecare componenta complexa si am ales tipul de date fractionar Word32, deoarece va stoca o valoarea rezultata in urma unei inmultiri cu acumulare, operatie cu termeni de tipul Word16, iar in modul_maxim se va stoca valoarea maxima a modulului la patrat din fiecare subbanda.

Am utilizat doua bucle FOR: prima folosita pentru a parcurge toate cele 8 subbenzi, in timp ce a doua sa parcurga toate cele 16 componente complexe din fiecare set.

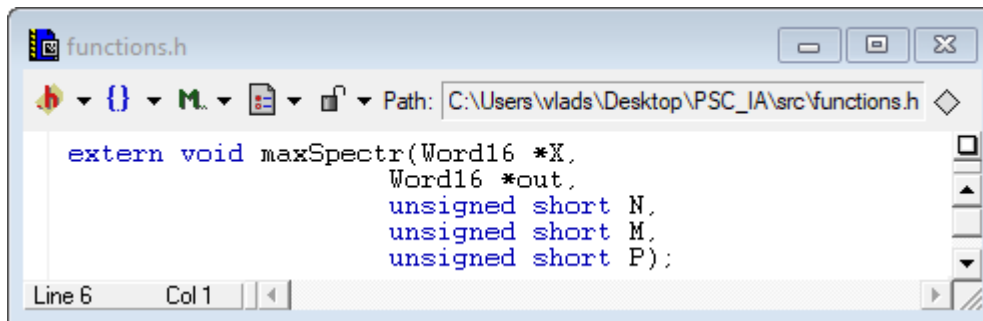
Pentru a ajunge la ecuatia modulului ridicat la patrat al unui numar complex, am folosit functia intrinseca L_mac, inmultirea cu acumulare, folosita pentru datele fractionare de tipul Word16 sau Word32.

Dupa ce am calculat modulul unei componente complexe, voi extrage cei mai semnificativi 16 biti din aceasta valoare utilizand functia extract_h() si voi atribui aceasta valoare variabilei modul_maxim.

Am folosit un algoritm simplu de comparatie, pentru a afla valoarea maxima din fiecare subbanda, urmand sa iterez variabila k astfel incat sa ajung la partea reala si cea imaginara a urmatoarei componente complexe.

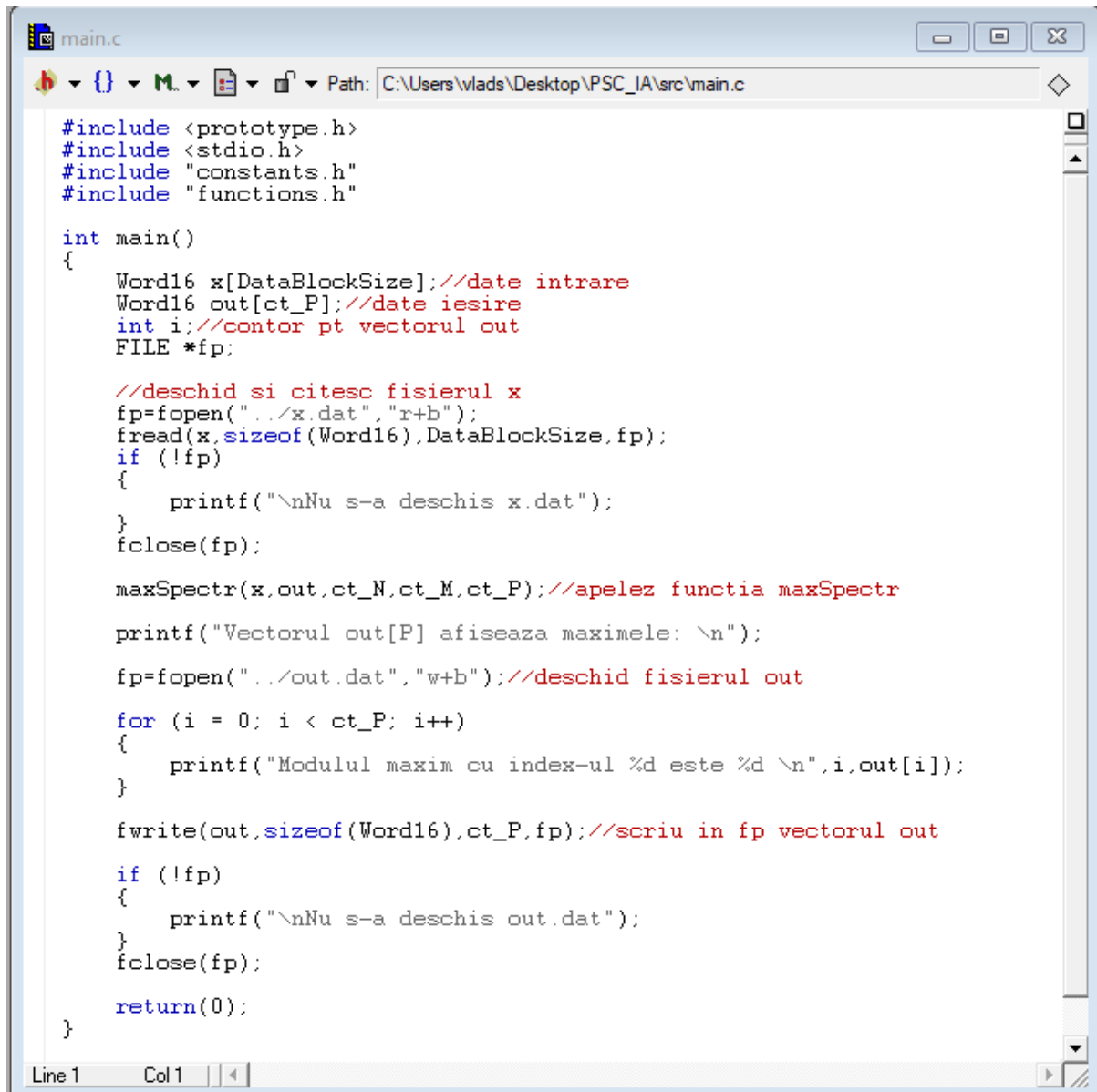


```
#define DataBlockSize 256 // vectorul X are 2N elemente
#define ct_N 128
#define ct_M 16
#define ct_P 8
```



```
extern void maxSpectr(Word16 *X,
                      Word16 *out,
                      unsigned short N,
                      unsigned short M,
                      unsigned short P);
```

Line 6 Col 1



```
#include <prototype.h>
#include <stdio.h>
#include "constants.h"
#include "functions.h"

int main()
{
    Word16 x[DataBlockSize]; //date intrare
    Word16 out[ct_P]; //date iesire
    int i; //contor pt vectorul out
    FILE *fp;

    //deschid si citesc fisierul x
    fp=fopen("../x.dat", "r+b");
    fread(x, sizeof(Word16), DataBlockSize, fp);
    if (!fp)
    {
        printf("\nNu s-a deschis x.dat");
    }
    fclose(fp);

    maxSpectr(x, out, ct_N, ct_M, ct_P); //apelez functia maxSpectr

    printf("Vectorul out[P] afiseaza maximele: \n");

    fp=fopen("../out.dat", "w+b"); //deschid fisierul out
    for (i = 0; i < ct_P; i++)
    {
        printf("Modulul maxim cu index-ul %d este %d \n", i, out[i]);
    }

    fwrite(out, sizeof(Word16), ct_P, fp); //scriu in fp vectorul out
    if (!fp)
    {
        printf("\nNu s-a deschis out.dat");
    }
    fclose(fp);

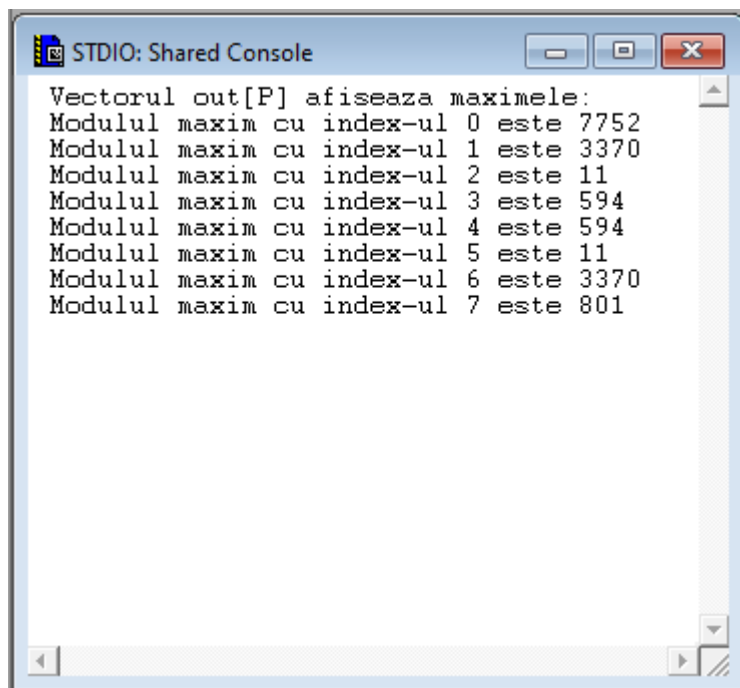
    return(0);
}
```

Line 1 Col 1

In functia **main** am inceput prin a modifica fisierele de intrare si de iesire corespondente cerintelor temei. Am deschis si am preluat valorile vectorului de intrare X rezultate in urma prelucrarii semnalului in Matlab. Am apelat functia **maxSpectr** pentru a prelucra din nou componentele vectorului si a rezulta vectorul de iesire out ce contine maximele patratelor modulelor.

Folosind sintaxa **fwrite** am realizat scrierea vectorului out, cu 8 valori de 16 biti.

Rezultatele afisate in consola:



```
Vectorul out[P] afiseaza maximele:  
Modulul maxim cu index-ul 0 este 7752  
Modulul maxim cu index-ul 1 este 3370  
Modulul maxim cu index-ul 2 este 11  
Modulul maxim cu index-ul 3 este 594  
Modulul maxim cu index-ul 4 este 594  
Modulul maxim cu index-ul 5 este 11  
Modulul maxim cu index-ul 6 este 3370  
Modulul maxim cu index-ul 7 este 801
```

4. Calculul erorii dintre valoarea rezultata in Matlab si cea obtinuta in Code Warrior

Am salvat intr-un vector numit „maxime_cw” cele 8 valori rezultate in urma prelucrarii in Code Warrior a vectorului X[].

```
fid=fopen('..\out.dat','r','b'); % deschiderea fisierului out.dat in care este stocat vectorul de iesire out[]  
maxime_cw = fread(fid,'int16'); % citirea valorilor  
maxime_cw = maxime_cw/(2^15); % valorile de tip fractionar Word16 din CW sunt transformate in valori fractionare  
fclose(fid); % inchiderea fisierului  
  
error = abs(maxime - maxime_cw'); % calculul erorilor
```

maxime								
1x8 double								
	1	2	3	4	5	6	7	8
1	0.2366	0.1028	3.5721e-04	0.0181	0.0181	3.5721e-04	0.1028	0.0245

maxime_cw		
8x1 double		
	1	
1	0.2366	
2	0.1028	
3	3.3569e-04	
4	0.0181	
5	0.0181	
6	3.3569e-04	
7	0.1028	
8	0.0244	

Se observa diferente foarte mici intre valorile celor 2 vectori.

error ✕								
1x8 double								
	1	2	3	4	5	6	7	8
1	1.2694e-05	4.2517e-06	2.1517e-05	1.6649e-05	1.6649e-05	2.1517e-05	4.2517e-06	1.2870e-05

Se observa ca erorile sunt foarte mici.