Ciele cvičenia

V priebehu cvičenia sa budeme zaoberať diskrétnou Fourierovou transformáciou (DFT) a pomocou princípu decimácie diskrétnej postupnosti v čase a vo frekvencii odvodíme algoritmus rýchlej Fourierovej transformácie. Odvodené algoritmy aplikujeme pri tvorbe a modifikáciach zdrojového kódu pre procesor TMS320C6713.

1. Úvod

Diskrétna Fourierova transformácia (DFT) umožňuje konverziu funkcie (diskrétnej postupnosti) definovanej v časovej oblasti na konečnom časovom intervale na ekvivalentnú funkciu (diskrétnu postupnosť) definovanú vo frekvenčnej oblasti na konečnom frekvenčnom intervale. Inverzná DFT umožňuje spätnú transformáciu diskrétnej postupnosti z frekvenčnej oblasti do časovej oblasti.
Rýchla Fourierova transformácia (FFT) je optimalizovaný algoritmus, ktorý vychádza z algoritmu DFT pričom využíva niektoré výhodné vlastnosti jeho symetrie.
Princíp dekompozície periodickej postupnosti pri výpočte DFT, ktorá je základom rýchlej Fourierovej transformácie, prvýkrát uviedli Cooley a Tukey v prelomovej práci z roku 1965. FFT je jedným z najčastejšie využívaných algoritmov v číslicovom spracovaní signálov, pretože umožňuje vykonávať spektrálnu analýzu signálov v reálnom čase. Pri odvodení algoritmu FFT je možné použiť dva rôzne prístupy: decimáciu v čase a vo frekvencii. V súčasnosti pojem FFT neoznačuje jediný algoritmus, ale skôr triedu algoritmov, takže sa môžeme stretnúť s množstvom modifikácií FFT, ako sú napríklad Winogradova transformácia, diskrétna kosínusová transformácia, alebo diskrétna Hartleyova transformácia.

2. Algoritmus FFT

Diskrétna Fourierova transformácia je definovaná dvojicou známych vzťahov v tvare konečných súčtov

$$X(k) = \sum_{n=0}^{N-1} x[n] W^{nk} \qquad k=0,1,2,...,N-1, \qquad (12.1)$$

kde vzorkovacia perióda T je zahrnutá v diskrétnej postupnosti x[n] a N je dĺžka tejto postupnosti. Konštanta W je obvykle v angloamericej literatúre označovaná “twiddle constant”, čo prekladáme ako faktor natočenia. Táto konštanta reprezentuje fázu, je definovaná

$$W = e^{-j\frac{2\pi}{N}}, \qquad (12.2)$$

a je funkciou dĺžky postupnosti N.
Vzťah (12.1) v skutočnosti reprezentuje výpočet hodnôt matice N×N, keďže k výpočtu X(k) potrebujeme určiť N hodnôt pre dané k. To znamená, že pre každé k je nutné vykonať (N-1) komplexných sčítaní a N komplexných násobení, t.j. celkovo potrebujeme pre N bodovú DFT (N^2-N) komplexných sčítaní a N^2 komplexných násobení. Z uvedeného vyplývajú značné výpočtové požiadavky už aj pre neveľké hodnoty N. FFT redukuje výpočtovú náročnosť z N^2 na Nlog N. Základom algoritmov FFT je výhodná vlastnosť periodicity a symetrie faktoru natočenia W.

Z periodicity W vyplýva

$$W^{k+N} = W^k, \qquad (12.3)$$

zo symetrie W vyplýva

$$W^{k+N/2} = -W^{k}. \tag{12.4}$$

Výhodná je aj nasledujúca vlastnosť W

$$W^{kN/2} = (-1)^{k}. \tag{12.5}$$

Úloha č. 1:
Dokážte vzťahy (12.3), (12.4) a (12.5)!

Princíp FFT spočíva predovšetkým v rozdelení *N*-bodovej DFT na dve (*N/2*)-bodové DFT. Každá (*N/2*)-bodová DFT je ďalej rozdelená do dvoch (*N/4*)-bodových DFT, atď. Posledná dekompozícia obsahuje potom (*N/2*) *2*-bodových DFT. Vychádzame z predpokladu, že dĺžka postupnosti *N* je 2^M, kde *M* je prirodzené číslo.

3. Algoritmus FFT na báze decimácie vo frekvencii

Rozdeľme pôvodnú postupnosť $x[n]$ dĺžky $N=2^M$ do dvoch postupností s polovičnou dĺžkou *N/2*. Pre DFT postupnosti $x[n]$ potom platí

$$X(k) = \sum_{n=0}^{(N/2)-1} x[n] W^{nk} + \sum_{n=N/2}^{N-1} x[n] W^{nk}. \tag{12.6}$$

Ak v druhej sumácií zavedieme substitúciu $n = n + N/2$, potom dostaneme

$$X(k) = \sum_{n=0}^{(N/2)-1} x[n] W^{nk} + W^{kN/2} \sum_{n=0}^{(N/2)-1} x\left[n + \frac{N}{2}\right] W^{nk}. \tag{12.7}$$

Po uvážení (12.5) môžeme predchádzajúci vzťah prepísať do tvaru

$$X(k) = \sum_{n=0}^{(N/2)-1} \left\{ x[n] + (-1)^{k} x\left[n + \frac{N}{2}\right] \right\} W^{nk}. \tag{12.8}$$

Vzhľadom na to, že $(-1)^k = 1$ pre párne hodnoty *k* a *-1* pre nepárne hodnoty *k*, môžeme vzťah (12.8) separovať do dvoch vzťahov

1. pre párne hodnoty *k*:

$$X(k) = \sum_{n=0}^{(N/2)-1} \left\{ x[n] + x\left[n + \frac{N}{2}\right] \right\} W^{nk}, \tag{12.9}$$

2. pre nepárne hodnoty *k*:

$$X(k) = \sum_{n=0}^{(N/2)-1} \left\{ x[n] - x\left[n + \frac{N}{2}\right] \right\} W^{nk}. \tag{12.10}$$

Zavedenie substitúcie $k = 2k$ pre párne hodnoty k a substitúcie $k = 2k + 1$ pre nepárne hodnoty k umožňuje písať rovnice (12.9) a (12.10) v tvare

$$X(2k) = \sum_{n=0}^{(N/2)-1} \left\{ x[n] + x\left[n + \frac{N}{2}\right] \right\} W^{2nk}, \tag{12.11}$$

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} \left\{ x[n] + x\left[n + \frac{N}{2}\right] \right\} W^{n} W^{2nk}. \tag{12.12}$$

Keďže faktor natočenia W je funkciou dĺžky postupnosti N, je obvyklé zapisovať ho v tvare W_N. Potom W_N^2 môžeme písať ako $W_{N/2}$. Zadefinujme nasledovné postupnosti

$$a[n] = x[n] + x\left[n + \frac{N}{2}\right], \tag{12.13}$$

$$b[n] = x[n] - x\left[n + \frac{N}{2}\right], \tag{12.14}$$

S využitím rovností (12.13) a (12.14) môžeme prepísať rovnice (12.15) a (12.16) do zrozumiteľnejšieho tvaru

$$X(2k) = \sum_{n=0}^{(N/2)-1} a[n] W_{N/2}^{nk}, \tag{12.15}$$

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} b[n] W_N^{n} W_{N/2}^{nk}. \tag{12.16}$$

Posledné dva uvedené vzťahy predstavujú dekompozíciu N-bodovej DFT do dvoch $(N/2)$-bodových DFT a tvoria tak bázu pre N-bodový algoritmus FFT.
Ak by sme takýmto spôsobom pokračovali v delení DFT, dospeli by sme k poslednej dvojbodovej dekompozícii v tvare

$$X(k) = \sum_{n=0}^{1} x[n] W^{nk} \qquad k = 0,1 \tag{12.17}$$

Uvedený algoritmus označujeme pojmom *decimácia vo frekvencii* (DIF), pretože výstupná sekvencia $X(k)$ je rozložená do menších subsekvencií. Je dôležité poznamenať, že FFT nie je aproximáciou DFT. Algoritmus FFT poskytuje rovnaké výsledky ako DFT pri podstatnej redukcii výpočtovej náročnosti. Táto redukcia je tým výraznejšia, čím je počet bodov DFT vyšší.

Úloha č. 2:
Nech x[n] je postupnosť pravouhlého okna definovaná nasledovne

x[n] = {1,1,1,1,0,0,0,0}.

Zostavte algoritmus 8-bodovej FFT s využitím decimácie vo frekvencii a vypočítajte hodnoty prvkov postupnosti X(k).
Výsledok vlastných výpočtov overte prostredníctvom programu Matlab.
Môžete použiť napríklad nasledujúci krátky m-file:

```
x = [1 1 1 1 0 0 0 0];
y = fft(x)
magy = abs(y)
plot(magy)
```

Podobne môžete overiť inverznú FFT:

```
X = [X(0) X(1) X(2) X(3) X(4) X(5) X(6) X(7)];
y = ifft(x)
```

3. Algoritmus FFT na báze decimácie v čase

Keďže proces decimácie vo frekvencii rozkladá výstupnú sekvenciu do menších subsekvencií, potom proces decimácie v čase (DIT) rozkladá do menších subsekvencií naopak vstupnú sekvenciu. Rozdeľme vstupnú diskrétnu postupnosť definovanú v čase na dve postupnosti, z ktorých jedna bude obsahovať len párne a druhá len nepárne členy pôvodnej postupnosti. Potom môžeme písať

$$X(k)=\sum_{n=0}^{(N/2)-1} x[2n]W^{2nk} + \sum_{n=0}^{(N/2)-1} x[2n+1]W^{(2n+1)k}. \tag{12.18}$$

Po úprave dostávame

$$X(k)=\sum_{n=0}^{(N/2)-1} x[2n]W_{N/2}^{nk} + W_N^k \sum_{n=0}^{(N/2)-1} x[2n+1]W_{N/2}^{nk}. \tag{12.19}$$

Výraz (12.19) reprezentuje dve (*N/2*)-bodové DFT.
Zadefinujme

$$C(k)=\sum_{n=0}^{(N/2)-1} x[2n]W_{N/2}^{nk} \tag{12.20}$$

$$D(k)=\sum_{n=0}^{(N/2)-1} x[2n+1]W_{N/2}^{nk}. \tag{12.21}$$

Vzťah (12.19) môžeme teraz zapísať v tvare

$$X(k)=C(k)+W_N^k D(k). \tag{12.22}$$

Rovnica (12.22) je definovaná pre $k \leq (N/2)-1$. Ak uvážime vlastnosť symetrie faktora natočenia (12.4), potom je možné definovať rovnicu (12.22) pre $k > (N/2)-1$ v tvare

$$X\left(k+\frac{N}{2}\right)=C(k)-W_N^k D(k). \tag{12.23}$$

Vzťahy (12.22) a (12.23) tvoria bázu pre algoritmus FFT založený na decimácii v čase.

Úloha č. 3:
Budeme sa zaoberať implementáciou algoritmu *N*-bodovej DFT na procesore TMS320C6713, ktorý spĺňa nasledovný predpis

$$X(k) = DFT\{x[n]\} = \sum_{n=0}^{N-1} x[n] W^{nk} \qquad \text{k = 0,1,2,...,N-1} \tag{12.24}$$

Rovnicu (12.24) môžeme rozdeliť na sumu reálnych zložiek a sumu imaginárnych zložiek

$$\mathrm{Re}\{X(k)\} = \sum_{n=0}^{N-1} x[n] \cos(2\pi nk / N), \tag{12.25}$$

$$\mathrm{Im}\{X(k)\} = \sum_{n=0}^{N-1} x[n] \sin(2\pi nk / N). \tag{12.26}$$

V nasledujúcom výpise krátkeho programu, ktorý realizuje výpočet DFT jednej periody funkcie kosínus doplňte telo funkcie, ktorá je jadrom výpočtu!

```
//DFT.c, N-bodova DFT udajov v poli.

#include <stdio.h>
#include <math.h>
void dft(short *x, short k, int *out);                  //prototyp funkcie
#define N 8                                              //pocet hodnot
float pi = 3.1416;

short x[N] = {1000,707,0,-707,-1000,-707,0,707};        //jedna perioda kosinusu
//short x[N]={0,602,974,974,602,0,-602,-974,-974,-602,
//             0,602,974,974,602,0,-602,-974,-974,-602}; //dve periody sinusu
int out[2] = {0,0};                                      //inicializacia vysledku

void dft(short *x, short k, int *out)                    //jadro transformacie
{
}

void main()
{
 int j;

 for (j = 0; j < N; j++)
   {
    dft(x,j,out);                                        //volanie funkcie DFT
   }
}
```

Po úspešnej kompilácii projektu DFT vykonajte nasledujúce kroky:

1. Vyberte View → Watch Window a zadajte dva výrazy *j* a *out*. Kliknite na *+out*, premenná typu poľa sa rozbalí, aby ste mohli sledovať hodnoty out[0] a out[1], ktoré reprezentujú reálnu a imaginárnu časť výsledku.
2. Umiestnite breakpoint na "{", ktorá nasleduje za volaním funkcie DFT.
3. Vyberte Debug → Animate (rýchlosť animácie je možné nastaviť v menu Options)

Získané výsledky overte pomocou Matlabu!
Projekt zopakujte s funkciou sínus v rozsahu dvoch periód!