2. Рудько Г. І. Дрогомирецький О. Я. Оцінка та прогноз зсувного процесу території Південно-Східного Передкарпаття // Матеріали конференції "Нормативно-правові аспекти оцінки і прогнозу екологічного стану довкілля адміністративних областей та регіонів України. – К., 1997. – С. 56-58.

3. Рудько Г.І. Дрогомирецький О.Я. Інженерно-геоморфологічний аналіз та розробка методики експертної прогнозної оцінки зсувів Південно-Східного Передкарпаття // Тези доповідей семінару "Застосування експертних методів при використанні екологічних досліджень". – К., 1996. – С. 42-44.

4. Емельянова Е.П. Сравнительный метод оценки устойчивости склонов и прогноза оползней. – М.: Недра, 1971. – 103 с.

5. Емельнянова Е.П. Основные закономерности оползневых процессов. – М.: Недра, 1972. – 310 с.

6. Ломтадзе В.Д. Инженерная геология. Инженерная геодинамика. – Л.: Недра, 1977.– 479 с.

7. Абрамович В.И. Изучение напряженно-деформированного состояния и устойчивости склонов. Инженерно-геологические прогнозы и моделирование: Учеб. пособие / И.П.Земильский, Е.А.Черкез, А.В.Гузенко и др. – Одесса: Одес. ун-т, 1983. – С. 88-104.

8. Григоренко А.Г. Измерение смещений оползней. – М.: Недра, 1988. – 144 с.

УДК 681.3.06(075)

ШВИДКИЙ АЛГОРИТМ ФІЛЬТРАЦІЇ ДЛЯ СИСТЕМ ДІАГНОСТУВАННЯ ОБ'ЄКТІВ НАФТОГАЗОВОГО КОМПЛЕКСУ

В.А.Ровінський, О.В.Євчук

ІФНТУНГ, 76019, м. Івано-Франківськ, вул. Карпатська, 15, тел. (03422) 47246
e-mail: k t s u @ n u n g . i f . u a

Предложен быстрый алгоритм цифровой фильтрации сигналов фильтрами с конечной импульсной характеристикой для Intel Pentium 4-совместимых процессоров, который обеспечивает максимальное быстродействие при произвольной длине импульсной характеристики.

The fast algorithm of digital signal filtration using FIR filters for Intel Pentium 4 compatible processors is proposed, which provides maximum processing speed for arbitrary impulse response length.

Цифровим системам обробки сигналів реального часу дії, що призначені для діагностування об'єктів нафтогазового комплексу, притаманна наявність алгоритмів фільтрації, зумовлена необхідністю якісного виділення інформаційного сигналу з шумових завад. Це стосуються практично всіх систем, які працюють з сигналами від реальних давачів: електронних динамографів та ватметрографів, що використовуються для діагностування штангових глибинно-насосних установок, ультразвукових дефектоскопів тощо. При цьому як обчислювальний засіб найчастіше використовують ЕОМ типу IBM PC, що зменшує собівартість розроблюваного пристрою, оскільки в такому разі не потрібно конструювати свій обчислювально-накопичувальний модуль системи діагностування. За умови зростаючих вимог до функціональності таких пристроїв нагальною є проблема вибору оптимального з точки зору швидкодії алгоритму фільтрації – алгоритму, який був би ефективним за будь-яких параметрів необхідного до реалізації фільтру.

Сучасні Intel 80x86-сумісні процесори Intel Pentium 4, AMD Athlon XP та їхні вдосконалені моделі здійснюють апаратну оптимізацію виконуваних програм, написаних для процесорів типу Intel 80386, при цьому за рахунок розпаралелювання виконання програмних інструкцій швидкість виконання програм з переходом прямо на нову модель процесора зростає вже не пропорційно до збільшення тактової частоти процесора, за рахунок чого значно збільшується швидкість роботи програми [1]. Тому поширені програмні середовища на зразок Borland Delphi або Microsoft Visual C++ з метою програмної сумісності генерують оптимальний код тільки для процесорів минулих поколінь. Проте можливість застосування асемблерних вставок в більшості сучасних мов програмування дає змогу досягти оптимальної ефективності коду і створити ефективний алгоритм фільтрації, придатний для практичного застосування в діагностичних системах, що працюють в реальному часі.

Принцип фільтрації вхідного сигналу за відомою імпульсною характеристикою фільтра описується відомим інтегралом згортки [2, 3]

$y(t) = \int_{-\infty}^{\infty} x(\tau) \cdot h(t-\tau) d\tau$, де $y(t)$ – вихідний сигнал; $x(\tau)$ – вхідний сигнал; $h(t)$ – імпульсна характеристика фільтруючого кола. Для випадку дискретних сигналів ця залежність

трансформується в: $y_i = \sum_{j=0}^{M} a_j x_{i-j}$ для фільтрів із скінченною імпульсною характеристикою та $y_i = \sum_{j=0}^{M} a_j x_{i-j} + \sum_{j=0}^{K} b_j y_{i-j}$ для фільтрів з нескінченною імпульсною характеристикою (тут M– довжина імпульсної характеристики в відліках, y_i – і-тий відлік вихідного сигналу). Очевидно, що час розрахунку за наведеними залежностями для неперервного вхідного сигналу залежить від довжини імпульсної характеристики, і за умови достатньо великого M час розрахунку може виявитися неприйнятним для систем реального часу дії. Тому існує більш швидкодіючий варіант згортки [4] з використанням швидкого перетворення Фур'є (англ. FFT-Fast Fourier Transform).

Алгоритм працює таким чином: імпульсна характеристика зі скінченною кількістю відліків переноситься з часової в частотну область шляхом застосування швидкого перетворення Фур'є. Аналогічна трансформація виконується і для вхідного сегменту сигналу, після чого здійснюється повідлікове взаємне перемножування дійсної та уявної частин імпульсної характеристики та сегменту вхідного сигналу. Одержані дійсна та уявна частини за допомогою зворотного швидкого перетворення Фур'є (IFFT) переносяться в часову область, і ми одержуємо вихідний сегмент сигналу. Операція множення, проведена у частотній області, повністю відповідає операції згортки в часовій області, однак при цьому внаслідок використання швидкого перетворення Фур'є значно економиться час для обчислень. Слід зауважити, що внаслідок відносної складності FFT-алгоритму виграш у часі від його застосування зростає при збільшенні числа відліків імпульсної характеристики. Як свідчать експериментальні дослідження [4], час, витрачений при проведенні фільтрації за допомогою простої згортки та FFT-згортки, залежить від кількості відліків імпульсної характеристики. Таким чином реалізуються цифрові фільтри зі скіченною імпульсною характеристикою, або FIR-фільтри (англ. Finite Impulse Response – скінченна імпульсна характеристика).

Враховуючи те, що практично часто вживані фільтри мають довжини імпульсних характеристик $N < 256$, було б корисно створити для них алгоритм, швидший, ніж FFT, використавши для цього розширений набір інструкцій SSE2 та SSE3 для процесорів типу Pentium III/4. Основна сутність застосування таких команд полягає в паралельному виконанні часовитратних операцій (множення, ділення, додавання та віднімання і т.д.) відразу для чотирьох 32-розрядних операндів, розміщених в спеціалізованих регістрах XMM0-XMM7.

Чотири 32-розрядні числа з плаваючою комою, розташовані в регістрах XMM0, XMM1, перемножуються і розміщуються в регістрі призначення XMM0 за схемою: $c1 = a1 \cdot b1$, $c2 = a2 \cdot b2$, $c3 = a3 \cdot b3$, $c4 = a4 \cdot b4$. Завантаження XMM-регістра з ОЗП здійснюється однією командою (наприклад **movups xmm0, [esi]**) значеннями, що розташовані послідовно в пам'яті. При цьому фактичний час виконання команди пересилки даних є співрозмірний із часом виконання множення. Тому використання XMM-команд не дає зменшення часу виконання FFT-алгоритму, а навіть навпаки, призводить до його зростання. Це пов'язано з тим, що вибірка даних для цього алгоритму здійснюється з пам'яті не послідовно (значення за значенням) а сітковим чином, причому відносна адреса розміщення даних кратна 2^i (тут i – крок робочого циклу).

Така схема потребує 5 операцій передачі даних – 4 послідовні операції передачі "пам'ять/пам'ять", коли дані з довільно розташованих комірок пам'яті переносяться в одну 128-розрядну змінну і стають послідовно впорядкованими. Після чого виконується команда завантаження **xmm** регістру **movups**, після якої можна застосувати команду пакетного множення **mulps** для XMM регістрів. Наявність додаткової команди пересилки даних перед пакетним множенням збільшує загальний час виконання порівняно із послідовною вибіркою/ перемножуванням даних, виконаних з використанням математичного співпроцесора за допомогою команди **fmul,** тобто у звичний спосіб. Тому використання XMM-команд таким чином для прискорення алгоритму швидкого перетворення Фур'є слід вважати неефективним.

У випадку використання звичайного алгоритму згортки можливе суттєве прискорення процесу при використанні SSE-команд за рахунок того, що дані обробляються послідовно і незалежно одні від одних, як це видно з реалізації цього процесу мовою Object Pascal:

```
procedure Filter ;
var i, j: longword;
begin
    for j:= M to XLen do
    begin // Згортка вхідного сигналу
            із імпульсною характеристикою
    Y[J]:= 0;
    for i:= 0 to M do
        begin
        Y[j]:= Y[j] + X[j-i] * H[i];
        end;
    end;
end;
```

Цей алгоритм може бути прискорений мінімум у чотири рази таким чином:

```
procedure FilterXMM;
var mm, nn, mmm, j : longword;
begin
    nn:=Xlen-M+1; mm:=M div 4;
    mmm:=M*4+4;
    for j:=0 to M do y[j]:=0;
```

```
    px:=addr(x[M+1]); py:=addr(y[M]);
    ph:=@h;
    //        Y[J]:= 0; FOR I:= 0 TO M do
              Y[J]:= Y[J] + X[J-I] * H[I];
    asm
    pushad
    movups xmm0,zero
    mov esi,px
    mov edi,py
    mov eax,ph
    mov ecx,nn
    @m1: movaps xmm1,xmm0      // y[i]:=0
    mov ebx,eax
    mov edx,mm
    @m2:   sub esi,16
    movups xmm2,[ebx]      // h[i]
    movups xmm3,[esi]      // x[j-i-3], [j-i-2],[
                           j-i-1],[j-i]
    shufps xmm3,xmm3,1bh       // x[j-i]
    mulps xmm3,xmm2        // h[i]*x[j-i]
    addps xmm1,xmm3        // y[j]+h[i]*x[j-i]
    add ebx,16
    dec edx
    jnz @m2
    haddps xmm1,xmm1       // SSE3-команда
    haddps xmm1,xmm1       // SSE3-команда
    movups xmm3,[esi-4]    // x[j-M]
    movups xmm2,[ebx]      // h[M]
    mulps xmm2,xmm3
    addss xmm1,xmm2
    movss [edi],xmm1
    add edi,4
    add esi,mmm // +(M+1)*4 -> return to x[j]
    loop @m1
    popad
end;
```

Три вищеописаних алгоритми (неоптимізована FIR-фільтрація, фільтрація з використанням SSE-команд та цифрова фільтрація з використанням FFT) було протестовано на вхідному файлі оцифрованого аналогового сигналу, записаного на протязі 1 хв. з частотою дискретизації 44100 Гц, тобто всього 2646000 відліків. Розрахунки проводились на комп'ютері з процесором Intel Celeron 4 з тактовою частотою 2,53ГГц (з підтримкою команд розширення SSE3). Результати тестування наведено на рис. 1.

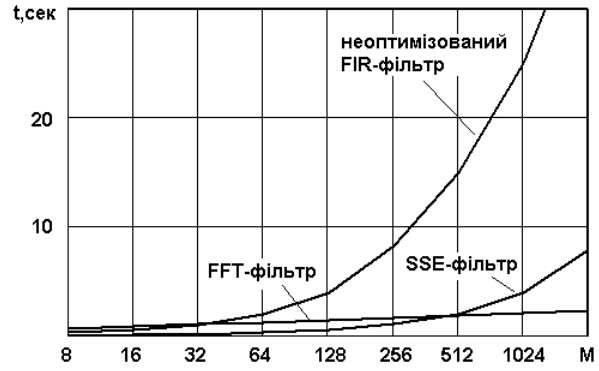

Рисунок 1 — Результати тестування алгоритмів згортки (M – довжина імпульсної характеристики фільтра)

В наведеному програмному фрагменті використано команду розширення SSE3 haddps – т.зв. "горизонтальне додавання" операндів в одному XMM-регістрі. Для процесорів, що не підтримують SSE3, цей фрагмент заміняється послідовністю з шести команд розширення SSE2. При цьому швидкодія алгоритму дещо знижується – на 7-10% при малих довжинах імпульсної характеристики фільтра (N=8...64) та на 1-5% для більших значень N.

Графічно запропонований алгоритм цифрової фільтрації наведений на рис.2.

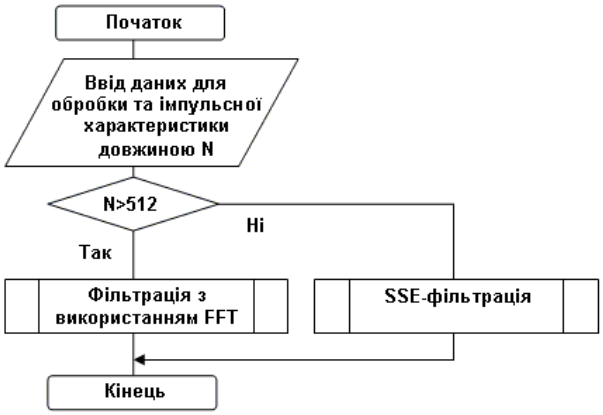

Рисунок 2 — Блок-схема алгоритму цифрової фільтрації даних

Отримані результати свідчать, що підвищення швидкодії при застосуванні SSE-команд порівняно з неоптимізованим алгоритмом складає до 1 секунди для довжини імпульсної характеристики N=8...128 та десятки секунд для великих N. За рахунок цього згортка за класичним алгоритмом з використанням SSE-команд виконується швидше, ніж FFT-фільтрація для всіх N<512, в той час як для неоптимізованого алгоритму – лише при N<20. При застосуванні неоптимізованого FIR-фільтра при N>2048 час обробки сигналу наближається до часу його запису, що вказує на неможливість застосування даного алгоритму для обробки даних в реальному часі для фільтрів з високою добротністю. Таким чином, при N<512 доцільно застосовувати алгоритм з використанням SSE-команд, а при N>512 більш вигідним стає застосування алгоритму цифрової фільтрації на основі швидкого перетворення Фур'є.

Література

1. Юров В. Assembler. – СПб.: Питер, 2002. – 624 с.

2. Сергиенко А.Б. Цифровая обработка сигналов. – СПб.: Питер, 2003. – 604 с.

3. Saeed V. Vaseghi. Advanced digital signal processing and noise reduction. – John Wiley & Sons, 2000. – 473p.

4. Steven W. Smith. "The Scientist and Engineer's Guide to Digital Signal Processing". – San Diego, 1999.