УДК 32.973.26-018.2

ВИКОРИСТАННЯ ГРАФІЧНИХ ПРОЦЕСОРІВ NVIDIA ДЛЯ ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ ОБЧИСЛЕНЬ КОРОТКИХ ЗГОРТОК В ЗАДАЧАХ ТРАНСВЕРСАЛЬНОЇ ФІЛЬТРАЦІЇ

Євчук О.В., Ровінський В.А., Стрілецький Ю.Й.
Івано-Франківський національний технічний університет нафти і газу
вул. Карпатська, 15, м. Івано-Франківськ, 76010
ktsu@nung.edu.ua

В роботі розглянутий спосіб реалізації фільтрів зі скінченною імпульсною характеристикою (СІХ) за допомогою графічних процесорів NVIDIA. Показано, що використання спеціалізованої розділюваної пам'яті дозволяє суттєво підвищити швидкодію алгоритму простої згортки, реалізованого на основі технології CUDA.

In the article was considered finite impulse responce (FIR) filter design method, which uses NVIDIA graphical processor units. It is shown that the use of a specialized shared memory can significantly increase the speed of a simple convolution algorithm implemented on the CUDA technology.

Цифрова фільтрація сигналів в останні десятиліття набула свого широкого розповсюдження в задачах обробки аудіо- та відеоданих, а також в різноманітних розділах радіотехніки, медицини, технічної діагностики та керування, в зв`язку з широким поширенням цифрових обчислювальних засобів, та досягнення ними значної обчислювальної продуктивності. В останні роки цифрова фільтрація починає активно розширювати свою присутність навіть при обробці високочастотних сигналів, фільтрація яких раніше здійснювалась тільки за допомогою аналогових RLC-кіл та активних схем на їх основі. Перевагами застосування алгоритмів цифрової фільтрації є наявність пам'яті, що дає можливість багатократно фільтрувати записаний потік вхідних даних, значно покращуючи при цьому масогабаритні та енерговитратні показники апаратури фільтрації, та зменшуючи похибку фільтрування. Для прикладу, реалізація аналогових фільтрів (з характеристиками Бесселя, Батерворта, Чебишева і т.д.) за нормальних умов можлива тільки до порядку $n \leq 10$, і навіть при цьому доводиться застосовувати елементи високого класу точності 0,1%, та здійснювати трудомісткі операції налаштування кожного з взірців фільтра. Використання цифрових алгоритмів дозволяє здійснювати аналогічну фільтрацію даних з використанням фільтрів з порядком до $n \leq 56$ без особливих складнощів в їх реалізації. Цифрові фільтри можуть реалізовуватись двома основними способами - за допомогою рекурсивних, або фільтрів з нескінченною імпульсною характеристикою (умовні скорочення рос. БИХ, англ. IIR-фільтр), або фільтрів з скінченою імпульсною характеристикою (рос. КИХ, англ. FIR-фільтр). Перевагою рекурсивних фільтрів є незначне навантаження на процесор, що особливо виявляється при фільтрації низькочастотних складових за умови низьких порядків фільтрів. Недоліком слід вважати складні або ресурсовитратні алгоритми підбору коефіцієнтів фільтра, його схильність до втрати стійкості, що частково долається використанням каскадних схем за рахунок зменшення продуктивності, та неможливість одержати лінійну фазочастотну характеристику (ФЧХ) фільтра, що є особливо важливою проблемою при побудові пристроїв для професійного звукозапису, та при обробці відеосигналів. Перевагами трансверсальних фільтрів є можливість побудови довільної амплітудно-частотної характеристики (АЧХ) фільтра, його гарантована стійкість в роботі, можливість одержання лінійної ФЧХ, та простий в реалізації алгоритм побудови імпульсної характеристики (ІХ), або ядра фільтра. Крім того, трансверсальних фільтр дозволяє здійснювати фільтрацію з довільними імпульсними характеристиками, які можуть бути виміряні експериментально з виходу деякої лінійної системи, і тоді трансверсальний фільтр може бути використаний як імітатор такої системи. Наприклад, в звукотехніці трансверсальні фільтри широко застосовують для імітації акустичних характеристик концертних залів та подібних приміщень (відтворюється їх реверберація), мікрофонів та звуковідтворювальних систем за їх виміряними або одержаними в результаті ауралізації ІХ. Недоліком трансверсальних фільтрів є необхідність застосування згортки, яка є основою трансверсальної фільтрації, для реалізації операції в високопродуктивних обчислювальних засобах.

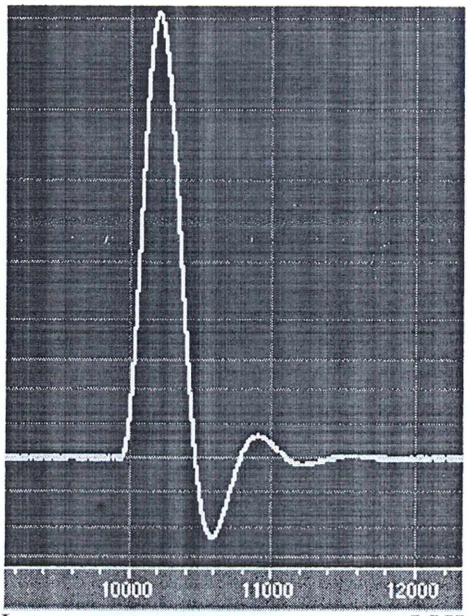

Рисунок 1 – Імпульсна характеристика ФНЧ Батерворта

Реалізація цифрових фільтрів може бути довільною. Для цього можна використати цифрові сигнальні процесори (ЦСП), або програмовані логічні матриці (ПЛМ) – в цьому разі вони несуть основне обчислювальне навантаження при фільтрації сигналів, однак розроблення таких засобів вимагає наявності додаткового обладнання, яке поступається в зручності використання розвиненим засобам розробки програмного забезпечення для персональних ЕОМ. Крім того, сама вартість ЦСП або ПЛМ підвищує вартість розроблюваної системи. Тому у випадку розробки стаціонарних систем, коли допустиме використання ПЕОМ, в якості обчислювальних засобів слід вибирати апаратне забезпечення персонального комп’ютера. Реалізація трансверсальних фільтрів проте суттєво залежить від довжини ядра фільтра, тобто від довжини його ІХ. В задачах ауралізації приміщень, тривалість ІХ (що фактично означає час затухання імпульсного звуку) для великих приміщень може бути 15сек і більше, що при частоті дискретизації 44100кГц складе 661500 відліків, що буде вимагати реалізації поблокових алгоритмів згортки, які є особливо складними при необхідності нульової затримки вихідного сигналу і виконуються з використанням алгоритму швидкого перетворення Фур’є (ШПФ). [1].

Водночас ІХ типового фільтра займає незначну кількість відліків. Для прикладу, на рис.1 наведена ІХ ФНЧ з характеристикою Батерворта (частота зрізу $f_0 = 74$ Гц, порядок фільтра $n = 4$, частота дискретизації сигналу $f = 44{,}100$ кГц).

З поданого на рис.1 видно, що ефективна довжина ІХ представленого фільтра складає 1800 відліків. В разі використання основного процесора для обчислень, для реалізації згортки доцільно вибрати алгоритм з використанням ШПФ, оскільки застосування багатопроцесорного варіанту реалізації такого алгоритму (для сучасних багатоядерних intel-процесорів)слід вважати «поганим тоном».

Проте існує можливість повністю обійтись без використання основного процесора, в разі якщо ПЕОМ обладнаний відеоплатою NVIDIA, з графічним процесором (GPU) який підтримує технологію Cuda. Список таких відеоплат досить широкий і наведений в табл.1 [2].

Таблиця 1

Відеоплати з підтримкою технології Cuda

Найменування	Найменування	Найменування	Найменування
GeForce GTX 480	GeForce 8800 Ultra	Quadro FX 3700M	Quadro CX
GeForce GTX 470	GeForce 8800 GTX	Quadro FX 3600M	Quadro NVS 450
GeForce GTX 295	GeForce 8800 GTS	Quadro FX 2700M	Quadro NVS 420
GeForce GTX 285	GeForce 8800 GT	Quadro FX 1700M	Quadro NVS 295
GeForce GTX 285 for Mac	GeForce 8800 GS	Quadro FX 1600M	Quadro NVS 290
GeForce GTX 280	GeForce 8600 GTS	Quadro FX 770M	Quadro Plex 2100 D4
GeForce GTX 275	GeForce 8600 GT	Quadro FX 570M	Quadro Plex 2200 D2
GeForce GTX 260	GeForce 8500 GT	Quadro FX 370M	Quadro Plex 2100 S4
GeForce GTS 250	GeForce 8400 GS	Quadro FX 360M	Quadro Plex 1000 Model IV
GeForce GT 220	GeForce 9400 mGPU	Quadro NVS 320M	Quadro FX 1800
GeForce G210	GeForce 9300 mGPU	Quadro NVS 160M	Quadro FX 1700
GeForce GTS 150	GeFórce 8300 mGPU	Quadro NVS 150M	Quadro FX 580
GeForce GT 130	GeForce 8200 mGPU	Quadro NVS 140M	Quadro FX 570
GeForce GT 120	GeForce 8100 mGPU	Quadro NVS 135M	Quadro FX 470
GeForce G100	GeForce GTX 280M	QuadroNVS 130M	Quadro FX 380
GeForce 9800 GX2	GeForce GTX 260M	Quadro FX 5800	Quadro FX 370
GeForce 9800 GTX+	GeForce GTS 260M	Tesla M2090	Quadro FX 370 Low Profile
GeForce 9800 GTX	GeForce GTS 250M	Tesla S2070	Quadro FX 5600
GeForce 9800 GT	GeForce GTS 160M	Tesla M2070	Quadro FX 4800
GeForce 9600 GSO	GeForce GTS 150M	Tesla C2070	Quadro FX 4800 for Mac
GeForce 9600 GT	GeForce GT 240M	Tesla S2050	Quadro FX 4700 X2
GeForce 9500 GT	GeForce GT 230M	Tesla M2050	Quadro FX 4600
GeForce 9400GT	GeForce 9700M GTS	Tesla C2050	Quadro FX 3800
GeForce GT 130M	GeForce 9700M GT	Tesla S1070	Quadro FX 3700
GeForce G210M	GeForce 9650M GS	Tesla S2090	
GeForce G110M	GeForce 9600M GT	Tesla C1060	GeForce 9200M GS
GeForce G105M	GeForce 9600M GS	Tesla C870	GeForce 9100M G
GeForce G102M	GeForce 9500M GS	Tesla D870	GeForce 8800M GTS
GeForce 9800M GTX	GeForce 9500M G	Tesla S870	GeForce 8700M GT
GeForce 9800M GT	GeForce 9300M GS		GeForce 8600M GT
GeForce 9800M GTS	GeForce 9300M G		GeForce 8600M GS
GeForce 9800M GS			GeForce 8400M GT
			GeForce 8400M GS

Використання графічного процесора для проведення математичних обчислень в загальному описане в [2,3,4,5,6,7,8]. Вибір способу реалізації алгоритму суттєво визначає час його виконання. Оскільки значну частину часу в даному алгоритмі займає читання відліків сигналу із глобальної пам'яті графічного процесора, чи не найголовнішу роль в забезпеченні реалізації швидкісного алгоритму відіграє правильне використання розділюваної пам`яті. Для сучасних

GPU кожний потоковий мультипроцесор містить 16кБайт розділюваної пам'яті (shared memory). Вона порівну розподіляється між всіма блоками сітки, що виконуються на мультипроцесорі. Крім того, розділювана пам'ять використовується також для передачі параметрів при запуску ядра на виконання, тому бажано уникати передачі значного об'єму вхідних параметрів, що безпосередньо передаються ядру в конструкції виклику ядра. При необхідності для передачі значного об'єму параметрів можна використати керованою константною пам'яттю. Дуже часто розділювана пам'ять використовується для зберігання постійно використовуваних значень – замість того, щоб постійно викликати з повільної глобальної пам'яті, зручніше їх один раз зчитати в розділювану пам'ять і в подальшому зчитувати їх тільки звідти. Такий спосіб дозволяє прискорити виконання програм для GPU у 5-6 разів, і досягти загального прискорення обчислень програми для GPU у порівнянні з програмами для CPU в 10-12 разів. В такому випадку можна суттєво скоротити витрати на звертання до глобальної пам'яті, якщо кожен із потоків у блоці читатиме лише два відліки сигналу: відлік з номером, що відповідає індексу потоку (який визначається через номери блоку та потоку), та відлік, затриманий на довжину ядра згортки.

Синхронізація потоків після операції читання за допомогою функції __syncthreads() гарантує, що на момент обчислення відліку згортки всі необхідні для цього відліки сигналу присутні в розділюваній пам'яті.

Функція обчислення згортки для GPU з використанням розділюваної пам'яті, виглядає наступним чином:

```
__global__ void gpuBruteConvol_constmem_shared(const float* X, float* Y, int
N, int M)
{
    __shared__ float xbuf[MAX_M];

    int k, m, k0;
    float sum=0;
    int j0,j1, i = blockDim.x * blockIdx.x + threadIdx.x;
    int delta=blockDim.x*gridDim.x;
    while(i < N+M-1)
    {
            j0=threadIdx.x+M-1;
            j1=threadIdx.x;
            if(i<N)
            {       k0=0;
                    if(threadIdx.x<M) xbuf[j1]=X[i-M+1];
            } else k0=i-N+1;
            if(i<M)
            {m=i+1;
            xbuf[j0]=X[i];
            } else m=M;

            __syncthreads();

            sum=0;
            for(k=k0; k<m; k++)   { sum+=xbuf[j0-k]*d_cH[k]; }
            Y[i]=sum;
            i+=delta;
    }
}
```

Тестування проводились на ПК з центральним процесором Core 2 Duo Q8200 (4 ядра, 2.3ГГц, 2Гб ОЗП) та графічною платою NVIDIA GeForce GT 9500 (4 процесорних ядра, 1.4 ГГц, 512 Мб ОЗП). Довжина ядра M приймалась рівною від 8 до 512, а довжина сигналу N – в межах від 10^4 до 10^6 відліків. Для визначення часу виконання коду на GPU застосовувались функції CUDA C для роботи з об'єктами типу cudaEvent_t, для CPU – функція Windows API QueryPerformanceCounter.

Був здійснений аналіз часу виконання пропонованого та різних реалізацій алгоритмів, описаних в [8]. Графічно результати експериментів по визначенню часу виконання для різних реалізацій алгоритму наведені на рис.2.

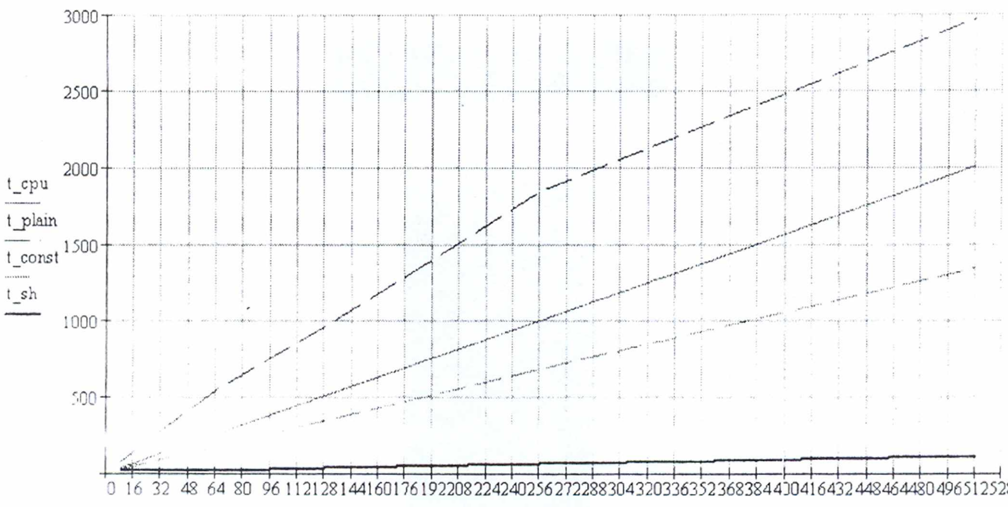

Рисунок 2– Час виконання згортки для різних довжин ядра n (8..512) при використанні різних підходів для реалізації алгоритму. Тут: t_cpu – час виконання алгоритму безпосередньо центральним процесором, t_const- алгоритм для GPU з використанням пам'яті констант, t_sh - алгоритм для GPU з використанням розділюваної пам'яті

Як видно з рис. 2, найменший час виконання має наведений для GPU алгоритм з використанням розділюваної пам'яті. Відносний приріст швидкодії у порівнянні з основним процесором наведений на рис. 3.

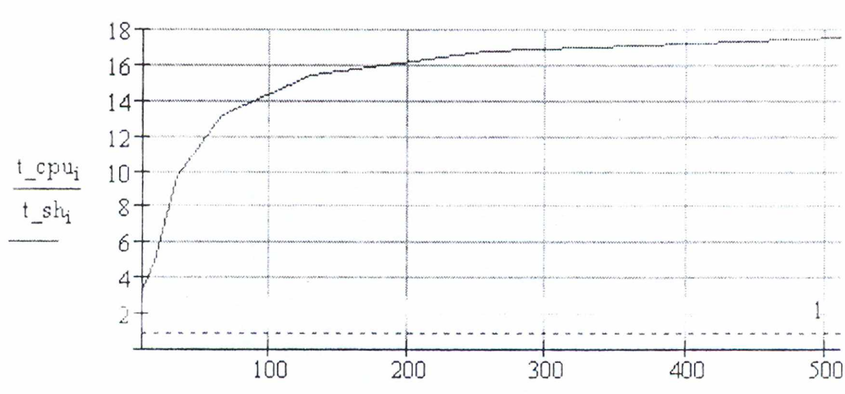

Рисунок 3– Відносне прискорення виконання алгоритму прямої згортки з використанням shared-memory алгоритму GPU по відношенню до прямого методу обчислення з використанням основного процесора

Отже, використання графічних процесорів дозволяє ефективно розпаралелити обчислення в алгоритмі дискретної згортки сигналу з ядром невеликого розміру. Достатньо значний приріст продуктивності порівняно з реалізацією на центральному процесорі, отриманий при тестуванні, пояснюється правильним способом вибору і використання розділюваної пам'яті. При використанні більш нових моделей GPU приріст повинен бути ще більш суттєвим.

Література

1. В.А. Ровінський, О.В. Євчук.. Швидкий алгоритм фільтрації для систем діагностування об'єктів нафтогазового комплексу // Розвідка та розробка нафтових і газових родовищ [Текст] : всеукр. наук.-техн. журн. - Івано-Франківськ, 2006. - N3. - С.63-65
2. Jason Sanders, Edward Kandrot. CUDA by example : an introduction to general-purpose GPU programming. Addison-Wesley, 2010. – 290pp.
3. Вычислительный процессор NVIDIA® TeslaTM C1060. // http://www.nvidia.ru/object/tesla_c1060_ru.html
4. NVIDIA CUDA C Programming Guide. Version 3.2 . – NVIDIA, 2010. // http://developer.download.nvidia.com/compute/cuda/3_0/toolkit/docs/ NVIDIA_CUDA_ProgrammingGuide.pdf
5. Michael Rush. "Convolution engine utilizing NVIDIA's G80 processor", 2007. - http://www.ece.ucdavis.edu/m~mrush/mmrush_eec277_final_writeup.pdf
6. 6.F.Wefers, J.Berg. High-performance real-time FIR-filtering using fast convolution on graphics hardware / Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-10), Graz, Austria , September 6-10, 2010
7. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA.- М.: ДМК Пресс, 2010.-232с., с ил.
8. О.В.Євчук, В.А.Ровінський,Ю.Й.Стрілецький. Використання графічних процесорів в задачах цифрової обробки сигналів в реальному часі // Методи і прилади контролю якості, 2010. - №25. - С. 97-100.

Стаття надійшла до редакційної колегії 23.09.2010 р.
Рекомендована до друку професором
Петришиним Л.Б.

УДК 62-83:622.24.05

ДОСЛІДЖЕННЯ ВИТРАТ ПОТУЖНОСТІ НА БУРІННЯ ТРІЩИНУВАТИХ ГІРСЬКИХ ПОРІД

Шаповал О.А., Кройтор Т.О.
Приватний вищий навчальний заклад «Галицька академія»
вул. Вовчинецька, 227, м. Івано-Франківськ, 76006

Рассмотрены вопросы повышения эффективности геологоразведочного бурения за счет исследования факторов, влияющих на работу электропривода буровой установки при бурении в трещиноватых породах. На основании оценки мощности, затрачиваемой на бурение, с учетом характера работы породоразрушающего инструмента и потерь мощности в колонне, определены основные требования к выбору электропривода. Произведен расчет мощности электродвигателя основного привода буровой установки.