

Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Локтионова Оксана Геннадьевна
Должность: проректор по учебной работе
Дата подписания: 09.02.2016 18:09:00
Уникальный программный ключ:
0b817ca911e6668abb13a5d426d39e5f1c11eabbf73e943df4a4851fda56d089

1

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

Юго-Западный государственный университет
(ЮЗГУ)

Кафедра вычислительной техники



ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ВИДЕОКАРТЫ С ПОДДЕРЖКОЙ ТЕХНОЛОГИИ CUDA В СРЕДЕ MICROSOFT VISUAL STUDIO

Методические указания по выполнению лабораторных и практических работ
для студентов направлений подготовки 09.03.01 и 09.04.01

УДК 621.3

Составитель: Э.И. Ватутин

Рецензент

Кандидат технических наук, доцент *А.Г. Сневаков*

Определение параметров видеокарты с поддержкой технологии CUDA в среде Microsoft Visual Studio: методические указания по выполнению лабораторной работы по дисциплине «Программирование распределенных и массивно-параллельных систем» / Юго-Зап. гос. ун-т; сост.: Э.И. Ватутин; Курск, 2016. 9 с.: ил. 2.

Методические рекомендации содержат сведения по разработке программных средств с использованием инструментария NVidia CUDA на современных языках программирования высокого уровня в среде Microsoft Visual Studio.

Предназначены для студентов направлений подготовки 09.03.01 и 09.04.01 «Информатика и вычислительная техника».

Текст печатается в авторской редакции

Подписано в печать _____ . Формат 60x84 1/16.

Усл. печ. л. Уч. – изд.л. Тираж 30 экз. Заказ . Бесплатно.

Юго-Западный государственный университет
305040, Курск, ул. 50 лет Октября, 94.

Содержание

Основные теоретические положения	4
Настройка среды Microsoft Visual Studio.....	4
Определение параметров видеокарты.....	6
Задание	7
Содержание отчета.....	8
Контрольные вопросы	9
Библиографический список	9

Основные теоретические положения

Цель работы: познакомиться с особенностями интеграции инструментария CUDA в среде разработки Microsoft Visual Studio, научиться определять основные параметры видеокарт, влияющие на скорость вычислений.

Настройка среды Microsoft Visual Studio

Инструментарий CUDA предоставляет возможность интеграции в состав популярной среды разработки Microsoft Visual Studio, что избавляет разработчика от необходимости напрямую обращаться к компилятору `nvc` с использованием командной строки или разрабатывать `make`-файлы. Для создания проекта с поддержкой CUDA необходимо (подразумевается, что инструментарий CUDA установлен на машине):

1. Создать новый проект.
2. Добавить к нему необходимые `.cu`-файлы.
3. Для каждого из них указать инструмент, с помощью которого производится компиляция (правый клик по файлу в дереве проекта → Properties → General → Tool → в выпадающем списке выбрать «CUDA Runtime API»):

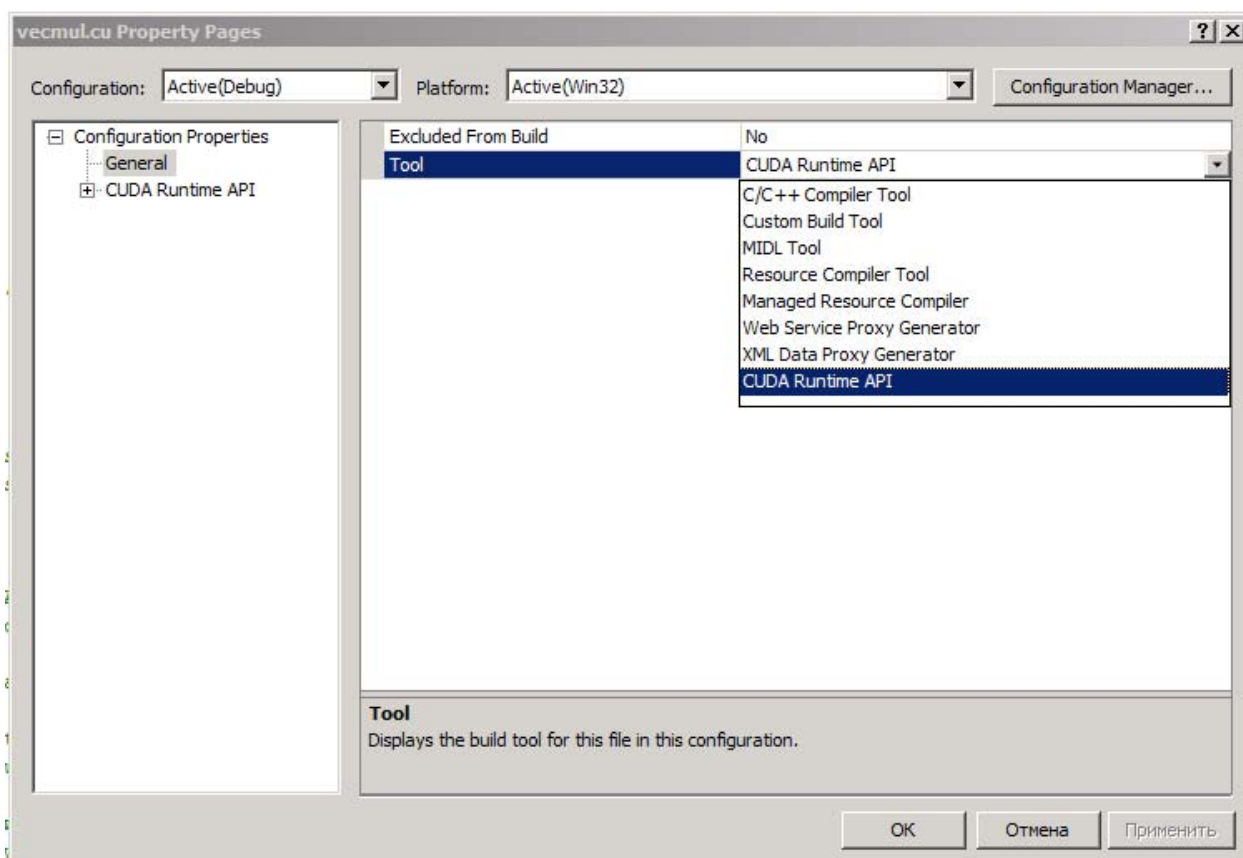


Рис. 1. Выбор типа компиляции для .cu-файлов

4. Указать путь к необходимым статически подключаемым библиотекам (выбрать Главное меню → Project → Properties → Configuration properties → Linker → Input → Additional dependencies, указать значение

```
"C:\Program Files\NVIDIA GPU Computing  
Toolkit\CUDA\v5.0\lib\Win32\cudart.lib"
```

(в кавычках, путь показан на примере Windows XP x86!)

Другой вариант – указать в коде программы ссылку на библиотеку:

```
#pragma comment(lib, "cudart.lib")
```

5. Указать путь к расположению инструментария CUDA (выбрать Главное меню → Project → Properties → Configuration properties →

Linker → Additional library directories, указать значение `$(CUDA_LIB_PATH)`.

6. Указать путь к расположению заголовочных файлов CUDA (выбрать Главное меню → Project → Properties → Configuration properties → C/C++ → Additional include directories, указать значение

"C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v5.0\include\"
(в кавычках, путь показан на примере Windows XP x86!)

После выполнения указанных действий среда разработки Microsoft Visual Studio будет самостоятельно вызывать компилятор `nvcc` для `.cu`-файлов, осуществлять компиляцию и сборку проекта.

Определение параметров видеокарты

Для определения числа установленных в системе видеокарт с поддержкой технологии CUDA используется функция `cudaGetDeviceCount()`:

```
int device_count;
cudaGetDeviceCount(&device_count);
```

Для определения свойств видеокарты используется функция `cudaGetDeviceProperties()`, принимающая в качестве параметра номер видеокарты и возвращающая в структуре `cudaDeviceProp` интересующие значения:

```
cudaDeviceProp dp;
cudaGetDeviceProperties(&dp, 0); // Определение параметров GPU с номером 0
```

Например, для определения наименований установленных в системе видеокарт с поддержкой CUDA и их вычислительных возможностей можно использовать следующий код:

```
int device_count;
cudaDeviceProp dp;

cudaGetDeviceCount(&device_count);
cout << "CUDA device count: " << device_count << "\n";

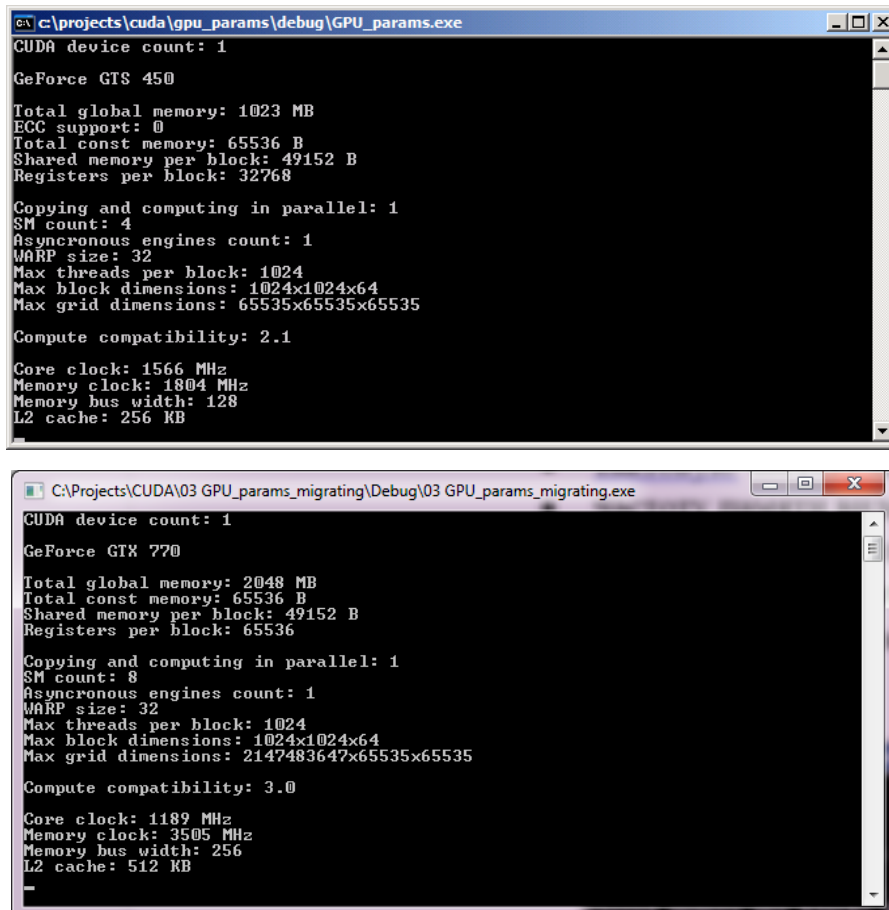
for (int i=0; i<device_count; i++)
{
    cudaGetDeviceProperties(&dp, i);

    cout << i << ": " << dp.name << " with CUDA compute compatibility " <<
        dp.major << "." << dp.minor << "\n";
}
```

Задание

1. Создать в среде Microsoft Visual Studio проект, состоящий из пары файлов (.cpp и .cu) из предыдущей работы. Установить в среде необходимые настройки. Убедиться в том, что проект успешно собирается и запускается.
2. Определить число видеокарт с поддержкой технологии CUDA и следующие параметры видеокарты:
 - наименование;
 - общий объем графической памяти;
 - объем памяти констант;
 - объем разделяемой памяти в пределах блока;
 - число регистров в пределах блока;
 - размер WARP'а;
 - максимально допустимое число потоков в блоке;
 - версию вычислительных возможностей;
 - число потоковых мультипроцессоров;
 - тактовую частоту ядра;

- частоту памяти видеокарты;
- объем кэша второго уровня;
- ширину шины памяти видеокарты;
- максимальную размерность при конфигурации потоков в блоке и блоков в сетке.



```
c:\projects\cuda\gpu_params\debug\GPU_params.exe
CUDA device count: 1
GeForce GTS 450
Total global memory: 1023 MB
ECC support: 0
Total const memory: 65536 B
Shared memory per block: 49152 B
Registers per block: 32768
Copying and computing in parallel: 1
SM count: 4
Asynchronous engines count: 1
WARP size: 32
Max threads per block: 1024
Max block dimensions: 1024x1024x64
Max grid dimensions: 65535x65535x65535
Compute compatibility: 2.1
Core clock: 1566 MHz
Memory clock: 1804 MHz
Memory bus width: 128
L2 cache: 256 KB

C:\Projects\CUDA\03 GPU_params_migrating\Debug\03 GPU_params_migrating.exe
CUDA device count: 1
GeForce GTX 770
Total global memory: 2048 MB
Total const memory: 65536 B
Shared memory per block: 49152 B
Registers per block: 65536
Copying and computing in parallel: 1
SM count: 8
Asynchronous engines count: 1
WARP size: 32
Max threads per block: 1024
Max block dimensions: 1024x1024x64
Max grid dimensions: 2147483647x65535x65535
Compute compatibility: 3.0
Core clock: 1189 MHz
Memory clock: 3505 MHz
Memory bus width: 256
L2 cache: 512 KB
```

Рис. 2. Основные параметры видеокарты, влияющие на производительность вычислений

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Задание.
4. Листинг программы.
5. Скриншоты с результатами работы программы.

6. ВЫВОДЫ.

Контрольные вопросы

1. Для чего предназначен инструментарий CUDA?
2. Как производится программирование NVidia GPU с использованием CUDA?
3. Как производится сборка программы для ее запуска на NVidia GPU?
4. Как инструментарий CUDA интегрируется в состав среды Microsoft Visual Studio?

Библиографический список

1. Емельянов С.Г., Ватулин Э.И., Панищев В.С., Титов В.С. Процедурно-модульное программирование на Delphi: учебное пособие. М.: Аргмак-Медиа, 2014. 352 с.
2. Зотов И.В., Ватулин Э.И., Борзов Д.Б. Процедурно-ориентированное программирование на C++: учебное пособие. Курск: КурскГТУ, 2008. 211 с.