Нижегородский государственный университет им. Н.И.Лобачевского

Мастер-класс по оптимизации с использованием Intel Parallel Studio

Мастер-класс по Intel Parallel Studio. Частотная фильтрация изображений (быстрое преобразование Фурье).

"Быстрое преобразование Фурье" - анализ, разработка, отладка, оптимизация, распараллеливание.

Сиднев А.А.
Кафедра математического обеспечения ЭВМ, факультет ВМК

14. Монстры

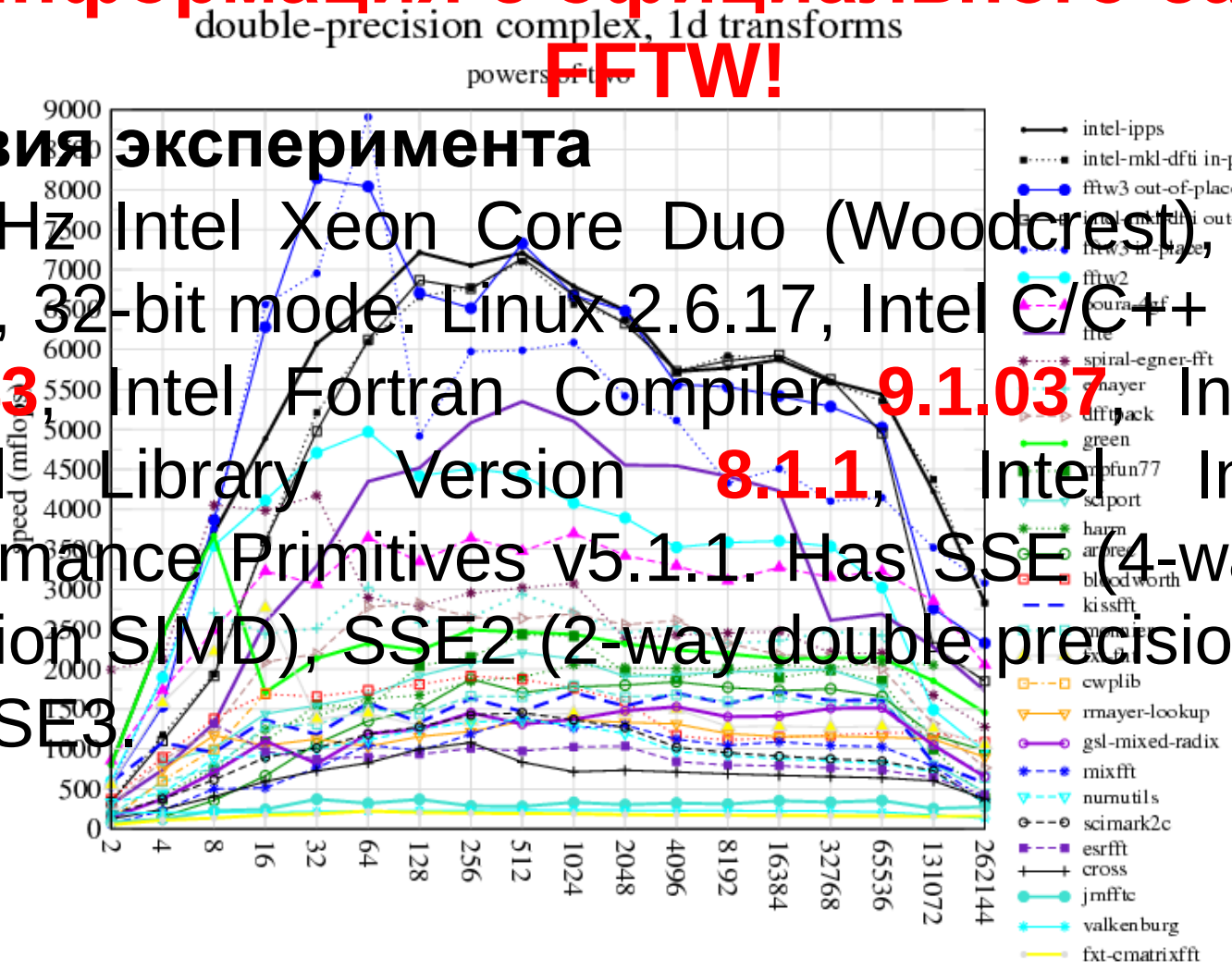
MKL для вычисления ДПФ

Реализация БПФ

- FFTW (Fastest Fourier Transform in the West)
  - библиотека для вычисления ДПФ;
  - GNU GPL.
- Intel MKL (Math Kernel Library)
  - содержит реализации большого количества алгоритмов;
  - оптимизирована для процессоров Intel;
  - имеет реализацию интерфейсов FFTW.

Сравнение FFTW и MKL

Информация с официального сайта FFTW!



Условия эксперимента

3.0 GHz Intel Xeon Core Duo (Woodcrest), 4MB L2 cache, 32-bit mode. Linux 2.6.17, Intel C/C++ Compiler **9.1.043**, Intel Fortran Compiler **9.1.037**, Intel Math Kernel Library Version **8.1.1**, Intel Integrated Performance Primitives v5.1.1. Has SSE (4-way single precision SIMD), SSE2 (2-way double precision SIMD), and SSE3.

Настройка среды разработки

- Путь к заголовочным файлам

  Tools → Options…→ Projects and Solutions → VC++ Directories

  Show directories for: **Include files**

  New Line: C:\Program Files\Intel\MKL\<version>\include

- Путь к статическим библиотекам для линковки

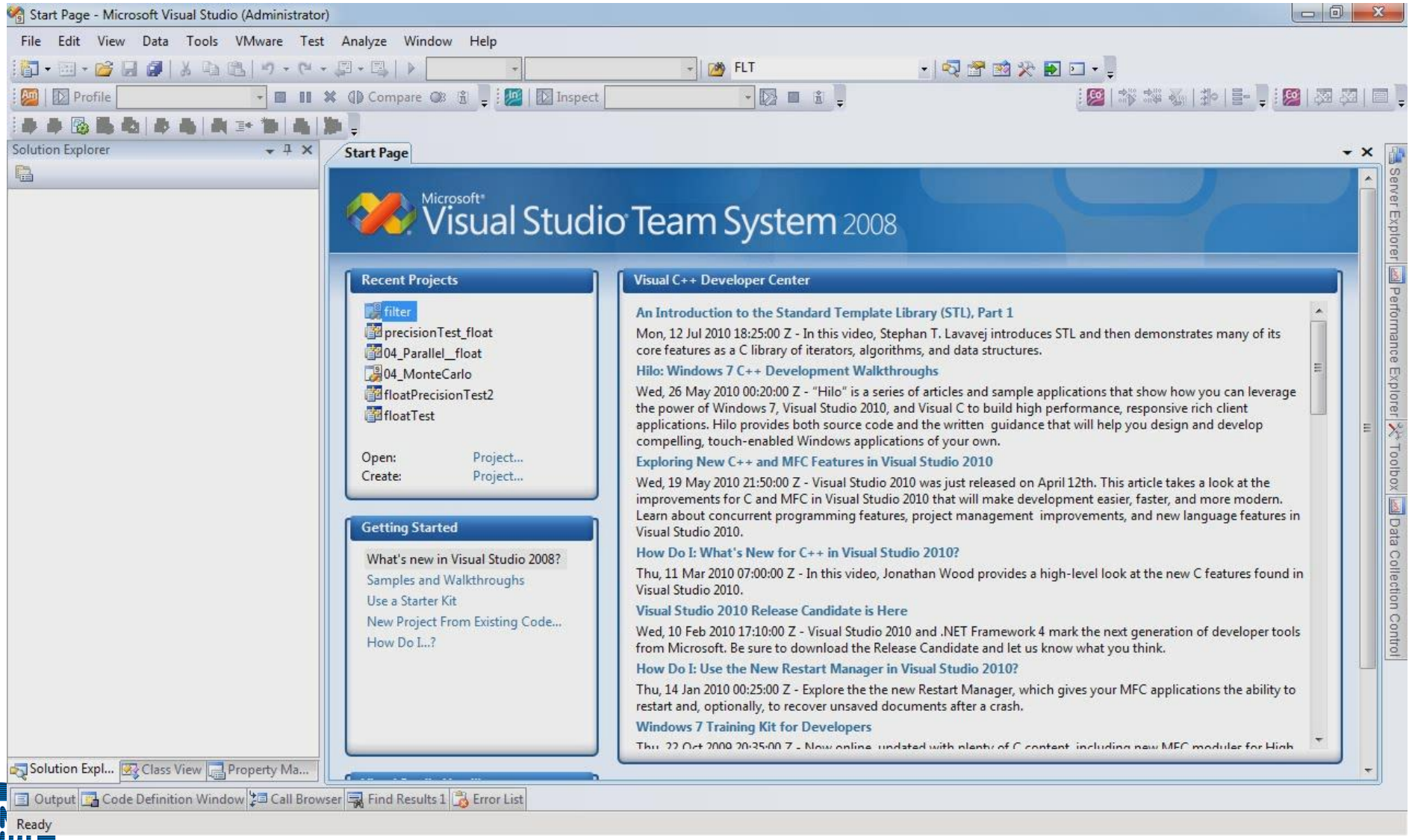
  Tools → Options…→ Projects and Solutions → VC++ Directories

  Show directories for: **Library files**

  New Line: C:\Program Files\Intel\MKL\<version>\<arch>\lib,

  где <arch> = ia32 | em64t | ia64

Настройка переменных окружения и среды разработки (демонстрация)



Использование ДПФ в MKL

- Заголовочный файл: **mkl_dfti.h**
- Статические библиотеки:
  - Базовая библиотека: **mkl_core.lib**
  - Версия MKL:
    - параллельная: **mkl_intel_thread.lib**
    - последовательная: **mkl_sequential.lib**
  - Тип интерфейса функций:
    - cdecl: **mkl_intel_c.lib**
    - stdcall: **mkl_intel_s.lib**
- Помощник для линковки: http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

Выполнение ДПФ в MKL

1. Создание дескриптора, описывающего ДПФ:
   `DftiCreateDescriptor`
2. Подтверждение дескриптора:
   `DftiCommitDescriptor`
3. Вычисление прямого/обратного ДПФ:
   `DftiComputeForward/DftiComputeBackward`
4. Освобождение дескриптора:
   `DftiFreeDescriptor`

Создание дескриптора

```
DFTI_DESCRIPTOR_HANDLE dftHandle;
MKL_LONG status;
status = DftiCreateDescriptor( &dftHandle, DFTI_DOUBLE,
   DFTI_COMPLEX, 1, size);
status = DftiCommitDescriptor( dftHandle );
```

- dftHandle – дескриптор.
- status – статус выполнения функции (status = 0, если операция выполнена успешно, используя функцию `DftiErrorMessage` можно получить содержательное сообщение о типе ошибки).
- DFTI_DOUBLE – тип элементов над которыми будет выполнять ДПФ – элементы двойной точности (может быть DFTI_SINGLE – одинарная точность).
- DFTI_COMPLEX – комплексная область (может быть DFTI_REAL – действительная область).
- 1 – размерность ДПФ.
- size – количество элементов.

Вычисление ДПФ

- Прямое ДПФ

```
status = DftiComputeForward( dftHandle, mas );
```

`mas` – последовательность действительных значений.

- Обратное ДПФ

```
status = DftiComputeBackward( dftHandle, mas );
```

`mas` – последовательность комплексных значений в формате.

Замечание

- При выполнении обратного ДПФ по умолчанию функция `DftiComputeBackward` **не** выполняет деление на количество элементов.

$$x_p = \frac{1}{n}\sum_{p=0}^{n-1} y_p e^{\frac{kp}{n}2\pi i},\ k = \overline{0, n-1}$$ - обратное ДПФ

- Можно реализовать ручное деление или перед выполнением операции `DftiCommitDescriptor` необходимо установить коэффициент масштабирования:

```
DftiSetValue(handle, DFTI_BACKWARD_SCALE,
                     1.0 / (double)n);
```

Освобождение дескриптора

```
status = DftiFreeDescriptor( &dftHandle );
```

Использование MKL (задание)

- Откройте проект **filter.sln**.
- Заменить ручное вычисление ДПФ на вычисление из MKL:
  - В файле **filter.cpp** функции **ParallelFFT** и **ParallelInverseFFT** выполняют вычисление ДПФ для каждой компоненты цвета. Для вычислений используются два массива для каждого канала (inp<R,G,B>, out<R,G,B>). Функциям MKL будет достаточно одного массива для каждого канала.
- Сравните быстродействие исходной версии и версии с MKL на различных видеофрагментах.