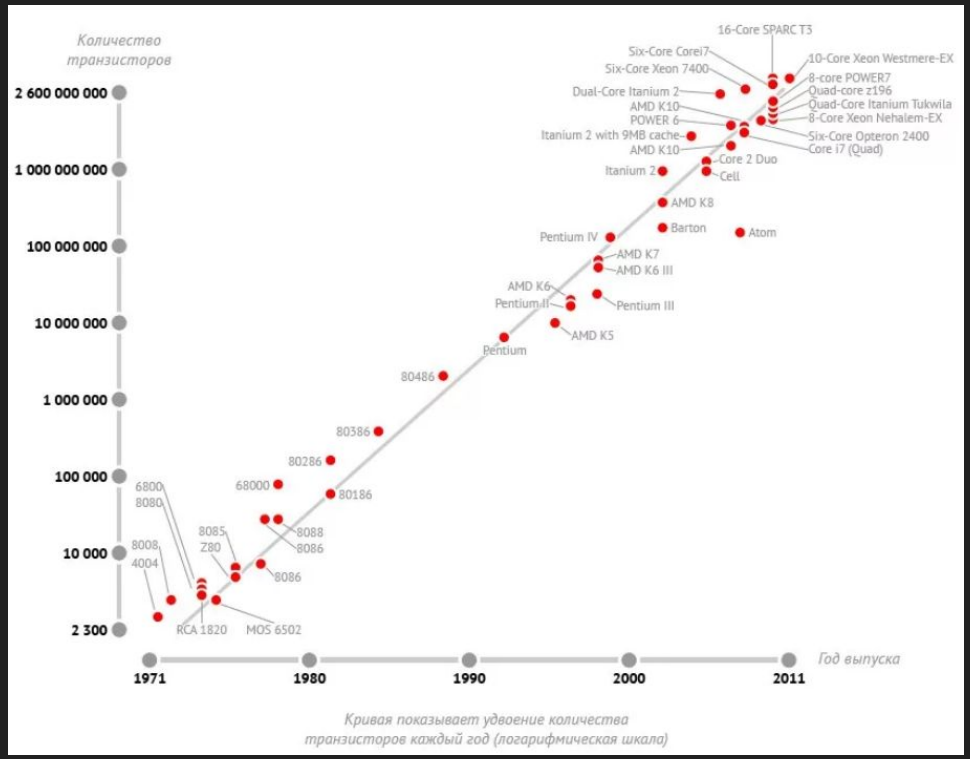
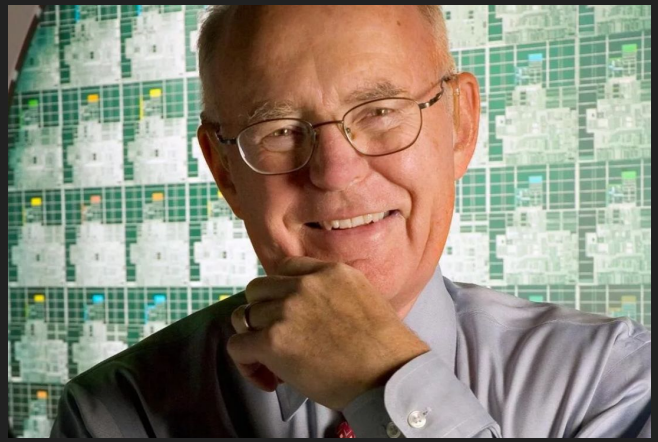
High-Performance Computing for Vision and Image Processing

Жилкин Федор, 444

Пролог. Закон Мура



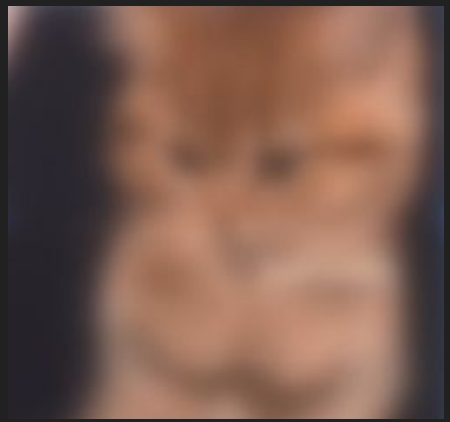


Преобразование Фурье

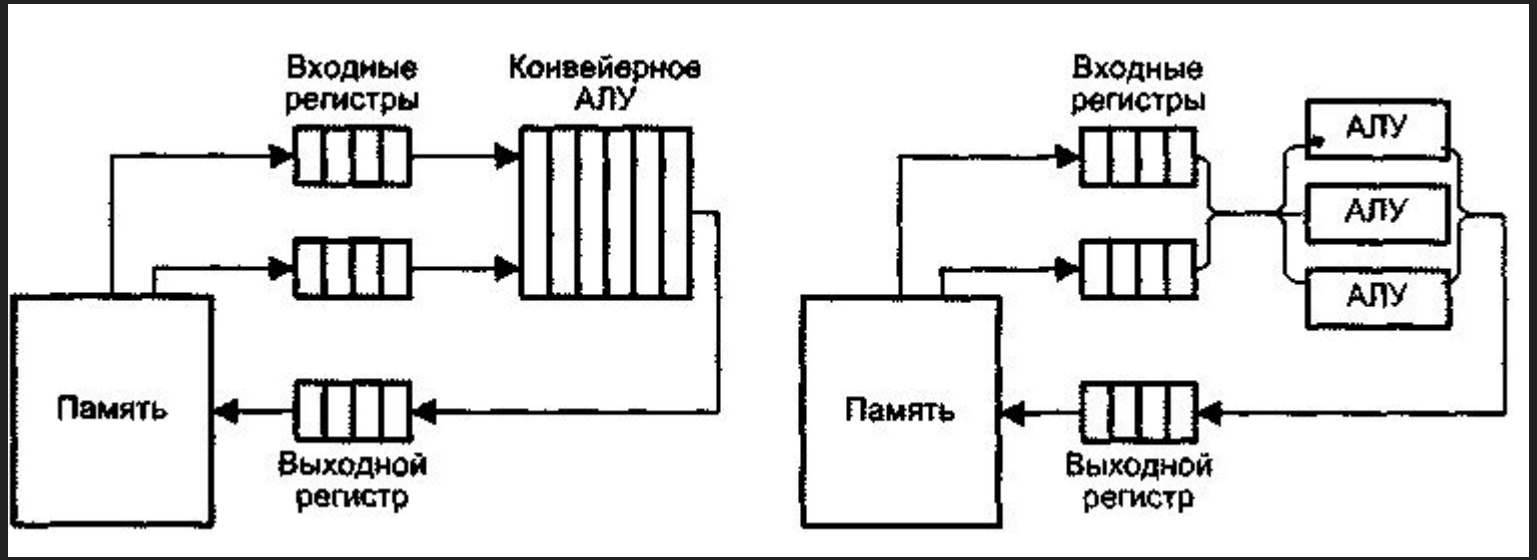
[Библиотека алгоритмов]

Алгоритм размытия изображения:

1. Пусть X(N1,N2) – массив яркостей пикселей изображения.
2. Вычислить Px = средняя (среднеквадратичная) яркость пикселей в массиве X
3. Вычислить массив Z=FT(X) – прямое двумерное дискретное преобразование Фурье
4. Вычислить массив Z′=T(Z), где T – обнуление строк и столбцов, находящихся в заданных внутренних областях матрицы-аргумента, соответствующих высоким частотам (то есть обнуление коэффициентов Фурье-разложения, соответствующих высоким частотам)
5. Вычислить массив Y=RFT(Z′) – обратное двумерное дискретное преобразование Фурье
6. Вычислить Py = средняя (среднеквадратичная) яркость пикселей в массиве Y
7. Нормировать массив Y(N1,N2) по среднему уровню яркости Px/Py



Парадигмы многопроцессорных вычислений. Векторный процессор



с конвейерным АЛУ

с несколькими АЛУ

Парадигмы многопроцессорных вычислений. Векторный процессор

Попарное сложение двух наборов по 10 чисел:

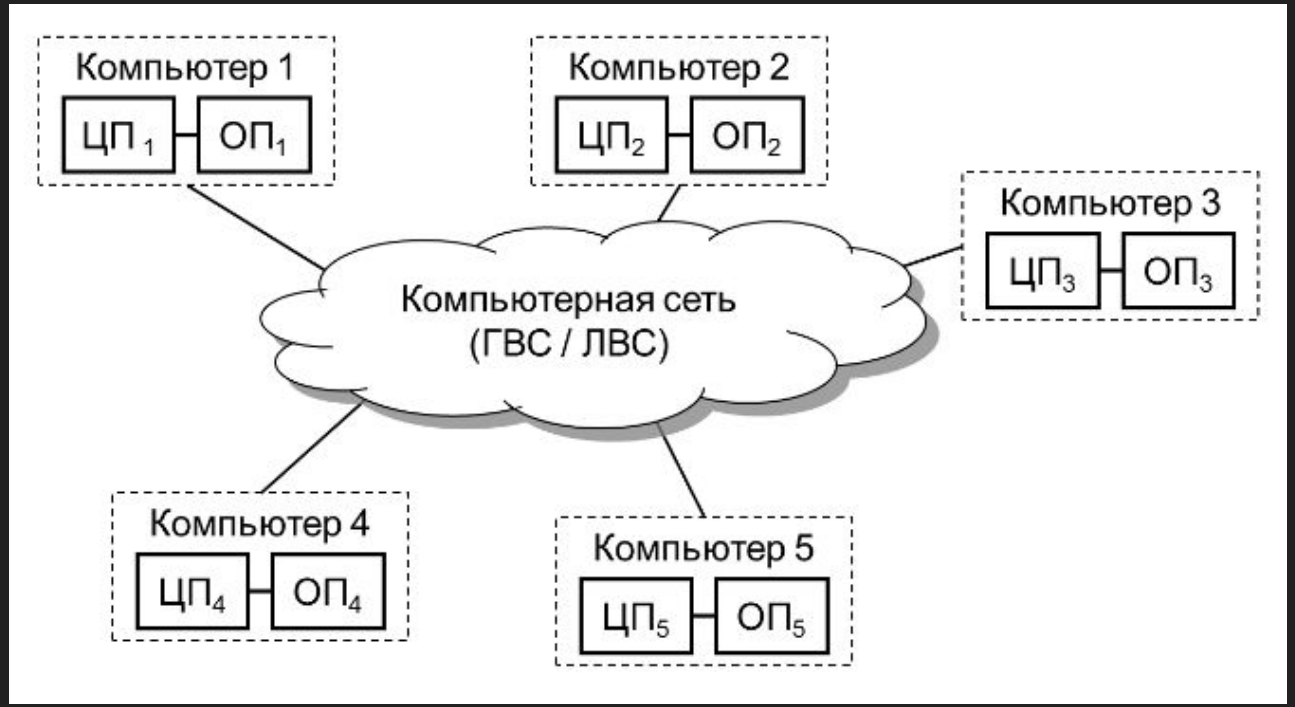
“Обычная” система:

повторить цикл 10 раз
    прочитать следующую инструкцию и декодировать
    получить первое слагаемое
    получить второе слагаемое
    сложить
    сохранить результат
конец цикла

Векторный процессор:

прочитать следующую инструкцию и декодировать
получить 10 первых слагаемых
получить 10 вторых слагаемых
сложить
сохранить результат

Парадигмы многопроцессорных вычислений. Распределённые вычисления



Разделяемая память

Message Passing

Про время исполнения

clock() -- C++

Stopwatch -- C#

<sys/time.h>, <chrono>

Gprof

QueryPerformanceCounter

QueryPerformanceCounter

(C++)

```
LARGE_INTEGER performanceCounter;
::QueryPerformanceFrequency(&performanceCounter);

LARGE_INTEGER performanceCounterStart;
::QueryPerformanceCounter(&performanceCounterStart);

DoWork();

LARGE_INTEGER performanceCounterEnd;
::QueryPerformanceFrequency(&performanceCounterEnd);

double timeElapsed = (double*)(performanceCounterEnd) - (double*)(performanceCounterEnd)
```

Message Passing Interface

Вычисление числа Пи с использованием **MPI:**

C

C#

Обработка изображений + MIP

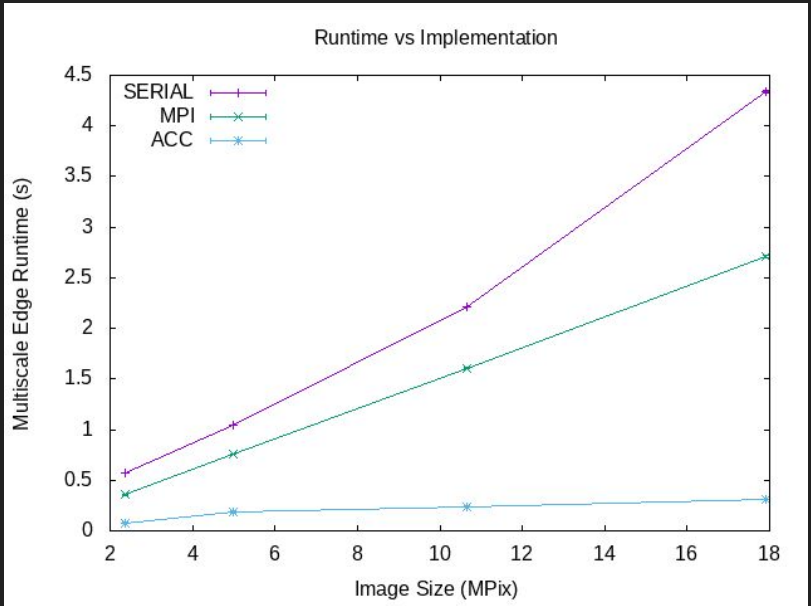
MPI_Send

```
int MPI_Send(void* message, int count,
    MPI_Datatype datatype, int dest, int tag,
    MPI_Comm comm)
```

https://github.com/Feodoros/ImageProcessingAlgorithms/blob/main/send_MPI.cpp

MPI using

https://github.com/YahiaBakour/multiscaleEdgeDetection/blob/master/mainMPI.cpp



Implementation	Runtime ± σ (s)	
Image Size	1920x1232	5184x3456
Serial	0.567 ±0.00078	4.34 ± 0.0075
ACC on Tesla	0.081 ± 0.0012	0.312 ± 0.00366
MPI-4 Cores	0.358 ± 0.004	2.711 ± 0.009
FFT Convolutions	91.23 ± 0.14	-----

Кластер

Кластер в IKEA шкафу:
http://helmer.sfe.se/

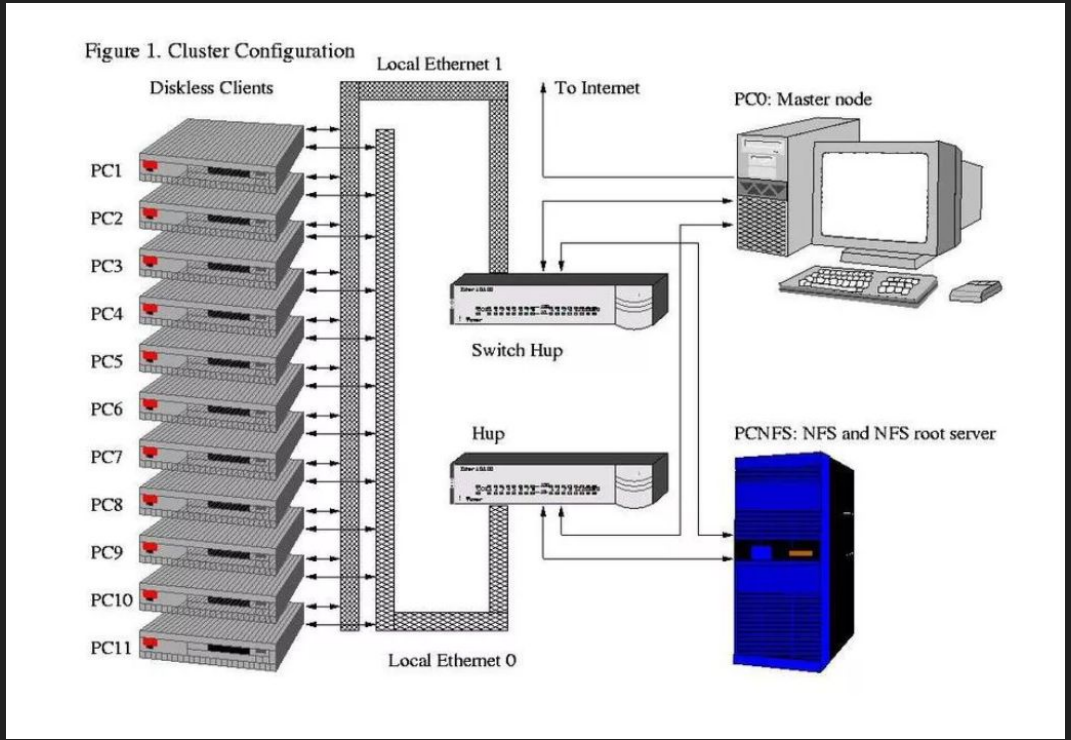
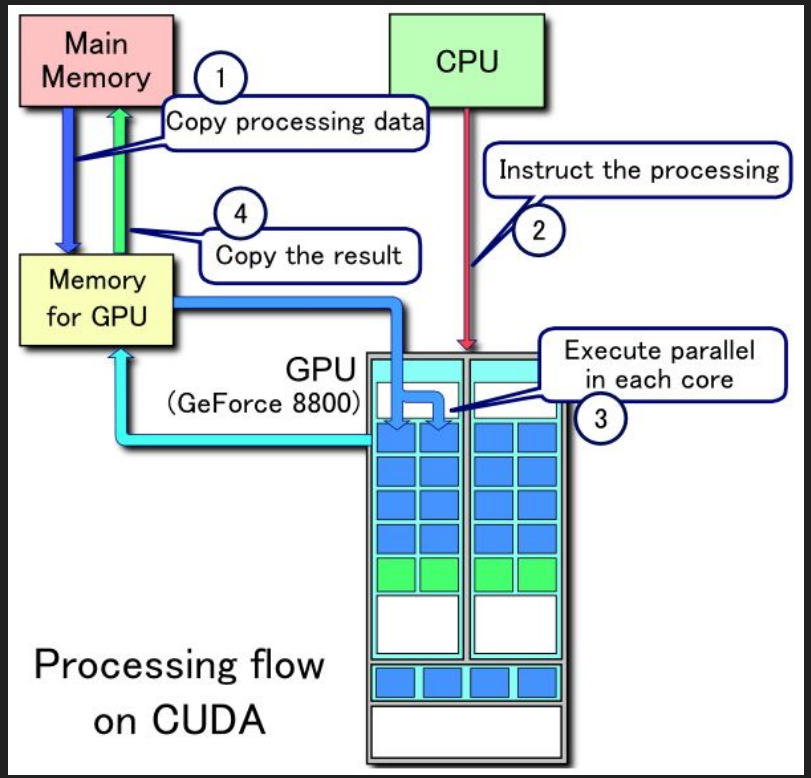


Image Processing on the GPU

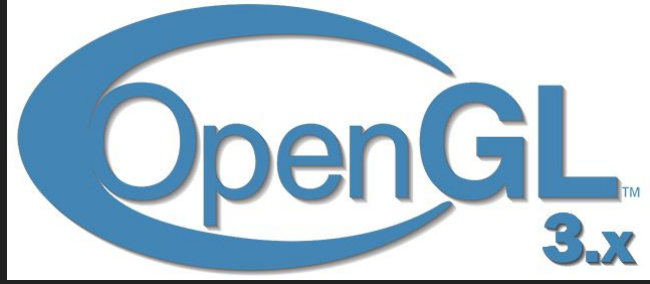
На примере CUDA

NVIDIA гайд

Пост о реализации ядра на ГПУ



GLSL



anyType variableName;

float f = 0.9;

vec4 v4 = vec4(f, 0.5, f, 1); // v4 = [0.9, 0.5, 0.9, 1]
vec3 v3 = vec3(v4); // v3 = [0.9, 0.5, 0.9]

const float ex1 = 1.5;
const vec4 ex2 = vec4(1, 0, 0, 0);
const vec4 ex2 = vec4(1, 2, vec2(0, 1));

[const]**convention** paramName

uniform (**type** | **samperType**) uniformName;

vec4 t0 = texture2D(uniform sampler2D mySampler2D, gl_TexCoords[0]);
vec4 t1 = textureCube(uniform samplerCube mySamplerCUBE, gl_TexCoords[1]);

Пример программы:
Вершинный шейдер
Фрагментный шейдер

Интересный пример