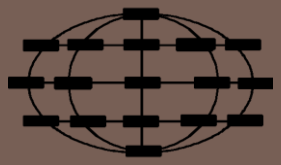
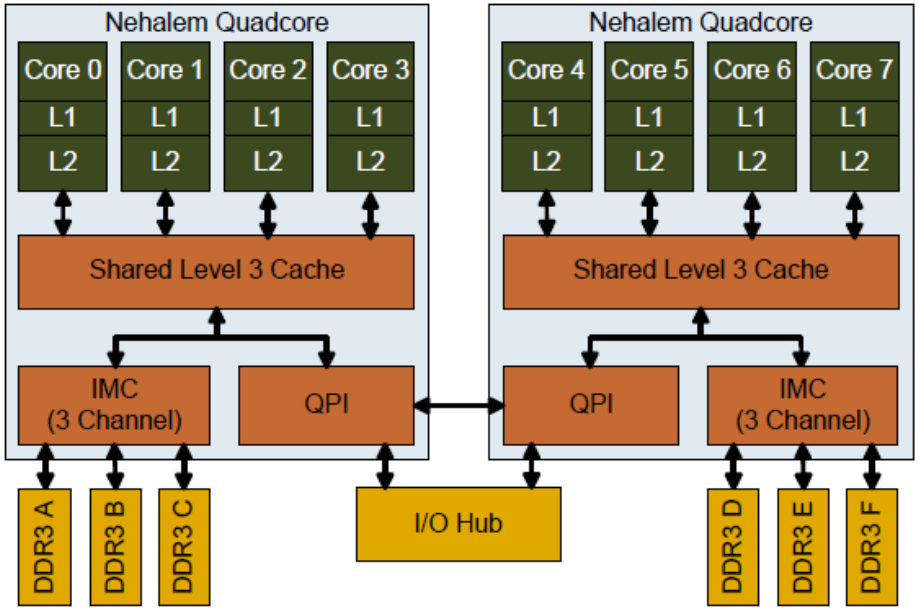
Институт Вычислительной Математики и Математической Геофизики
лаборатория Синтеза Параллельных Программы



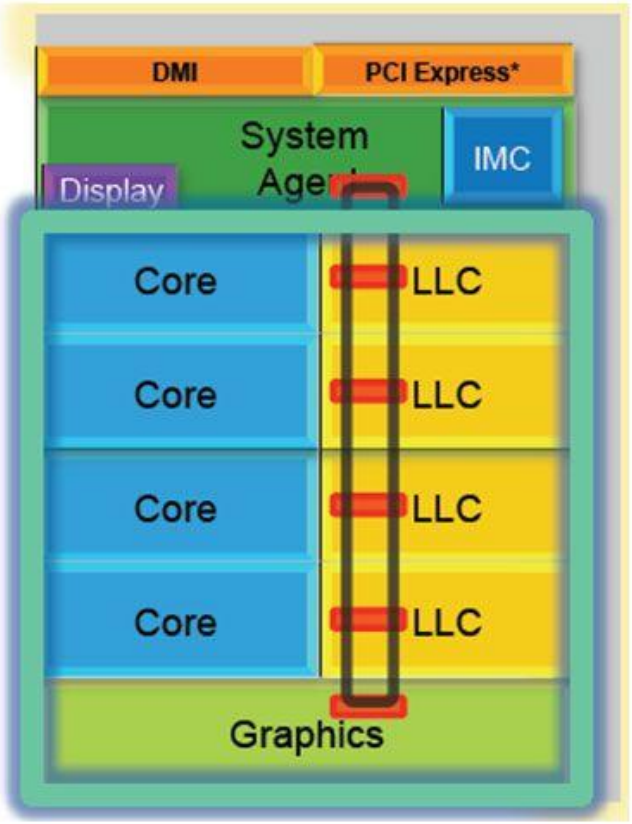
SMP, NUMA
Барьерная синхронизация

Константин Калгин
Киреев Сергей

Intel, ccNUMA

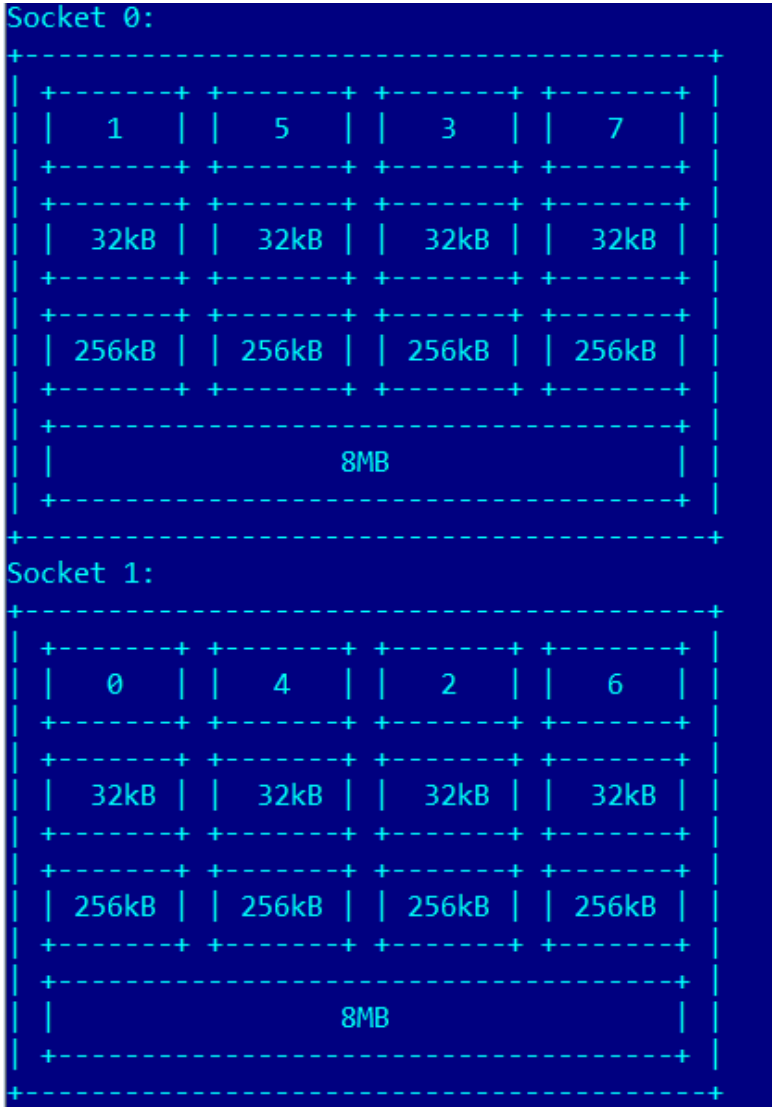


Nehalem/Westmere

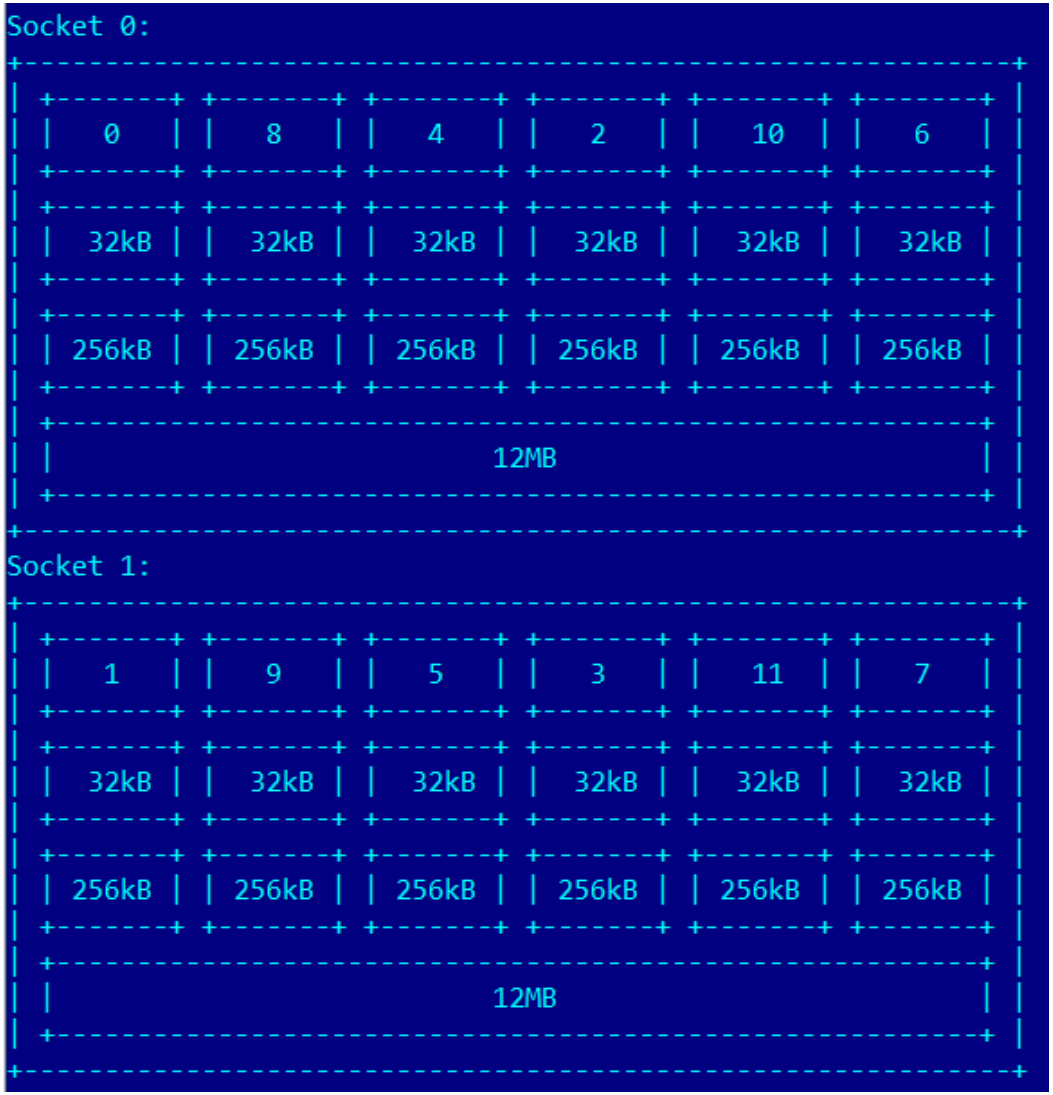


Sandy Bridge

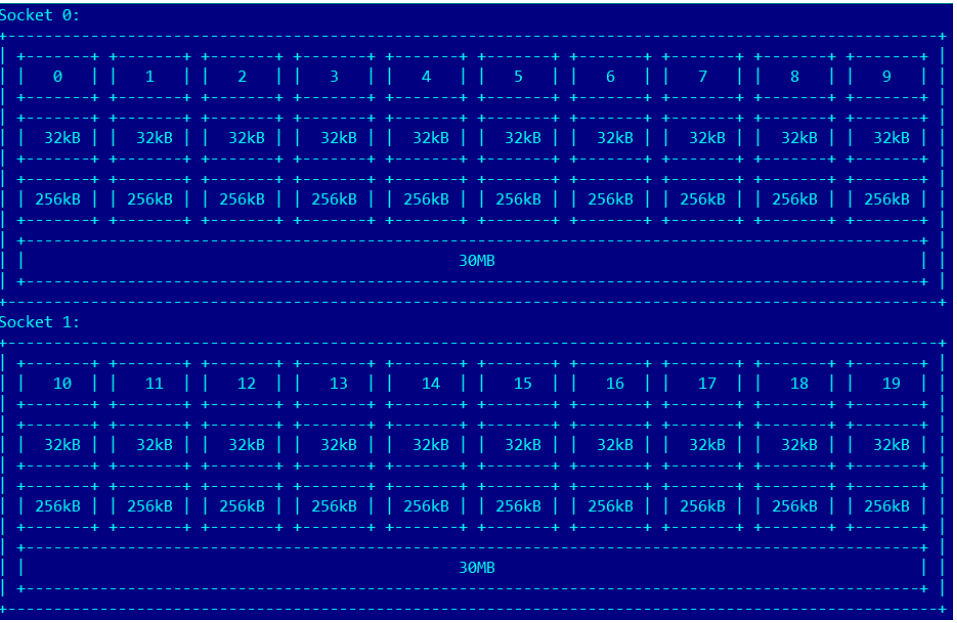
8 ядер, 12 Gb

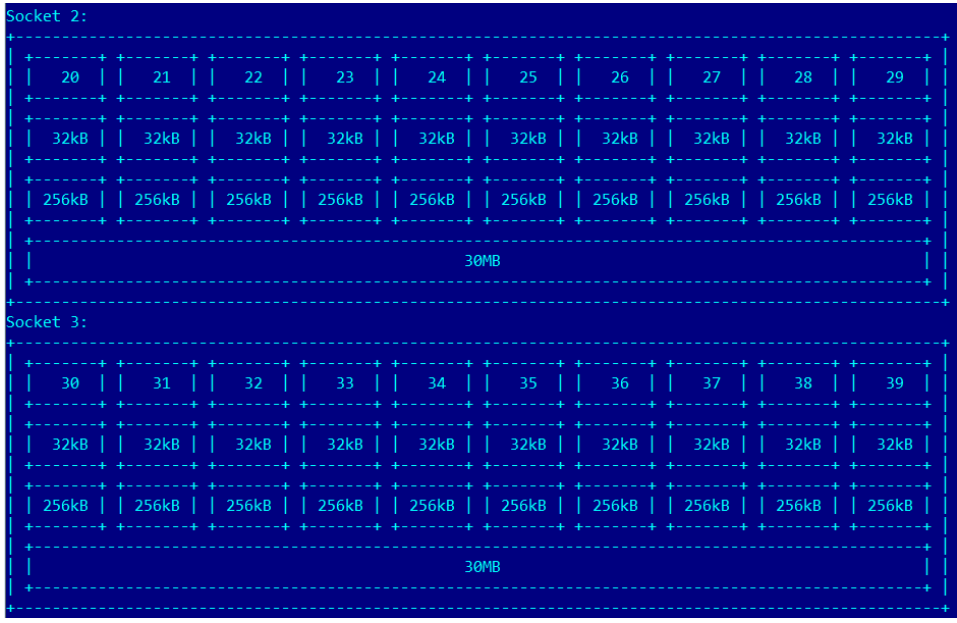

Socket 0:
1
5
3
7
32kB
32kB
32kB
32kB
256kB
256kB
256kB
256kB
8MB
Socket 1:
0
4
2
6
32kB
32kB
32kB
32kB
256kB
256kB
256kB
256kB
8MB

12 ядер, 48 Gb


Socket 0:
0
8
4
2
10
6
32kB
256kB
12MB
Socket 1:
1
9
5
3
11
7
32kB
256kB
12MB

40 ядер, 128 Gb





Latency, Magny-Cores, AMD

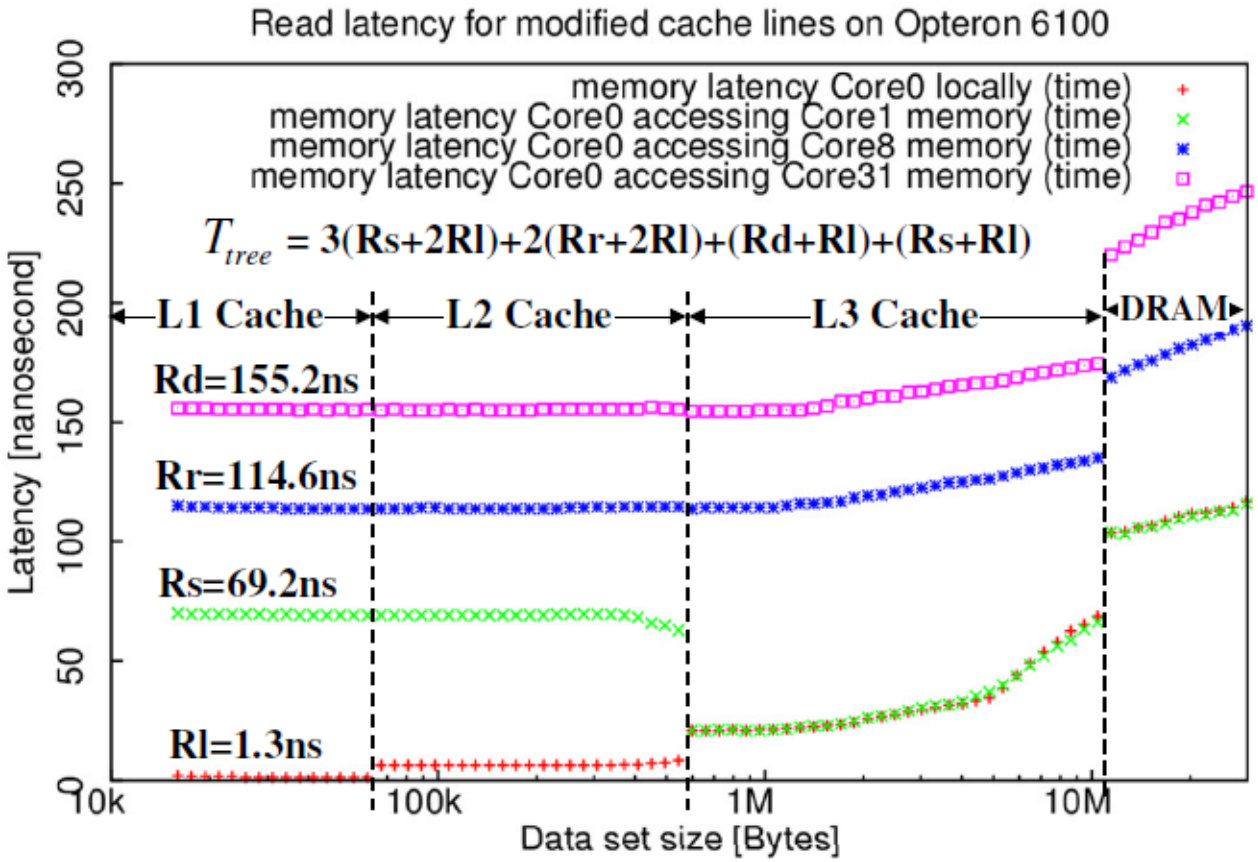


Figure 17: Modified cache line read latency on Magny-Cours. Rr denotes the L1 read latency from a horizontal or vertical remote socket, and Rd denotes the L1 read latency from a diagonal remote

Latency,Westmere, Intel

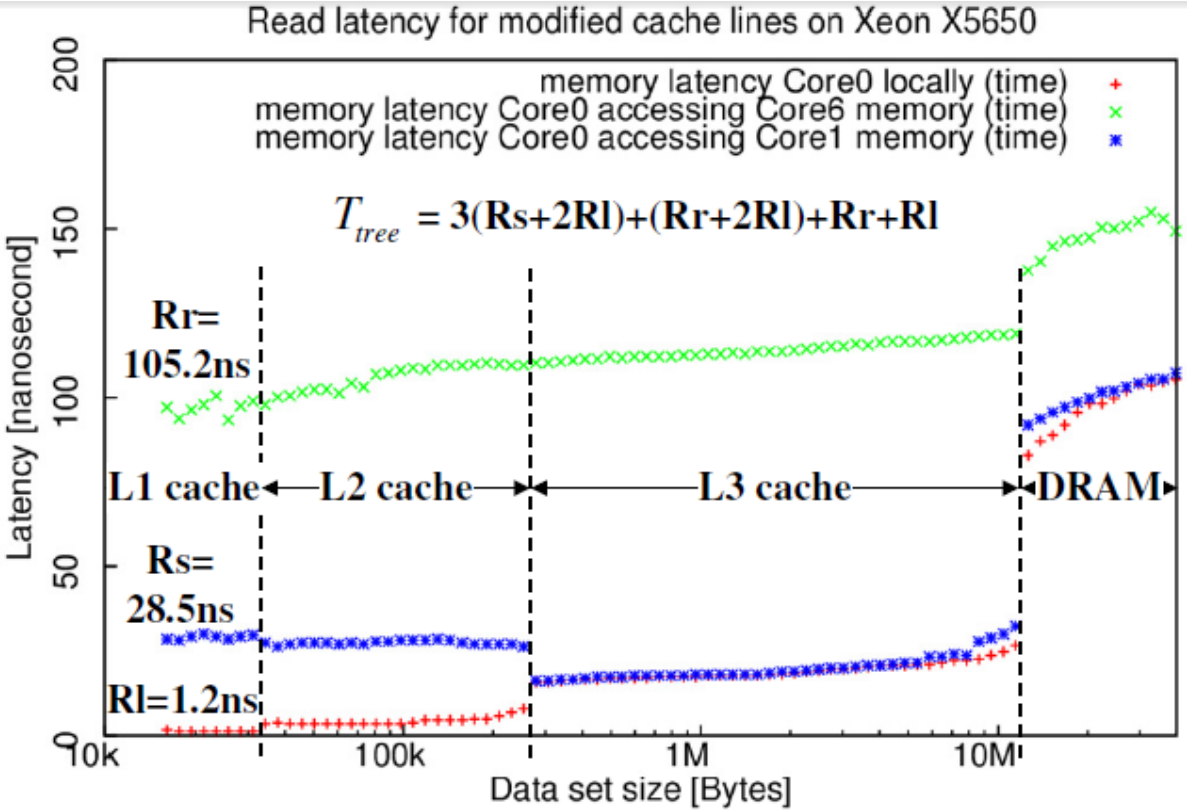


Figure 15: Modified cache line read latency on Westmere. Rl denotes latency of read local L1 cache, Rs denotes latency of read other L1 cache but within the same socket, and Rr denotes latency of read other L1 cache from remote socket.

Bandwidth, Wesmere, Intel

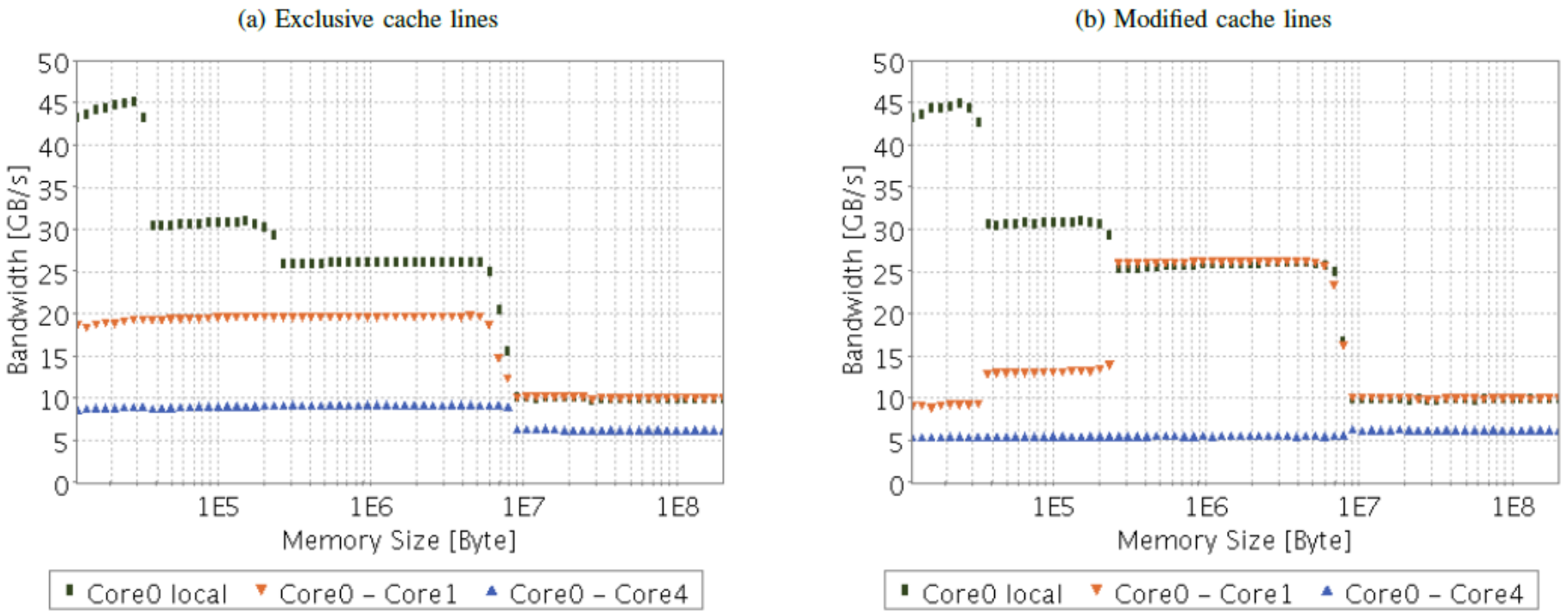
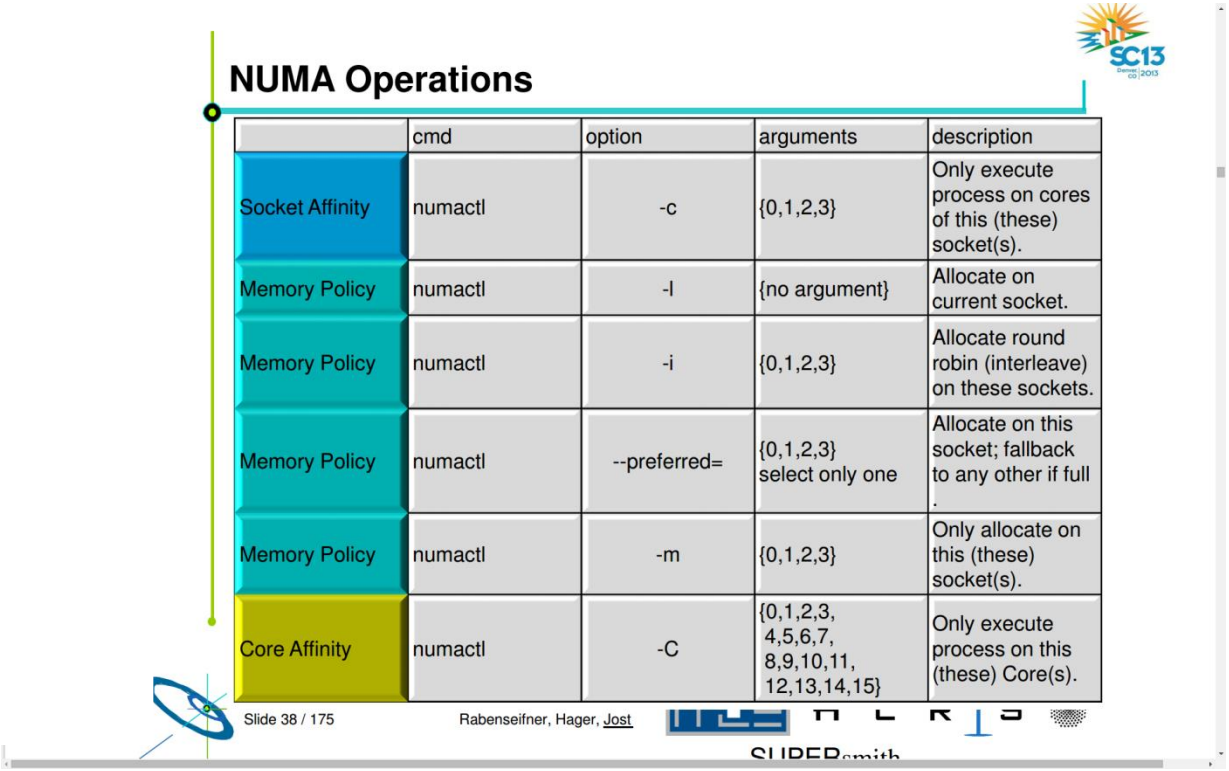


Figure 3. Read bandwidth of core 0 accessing cache lines of core 0 (local), core 1 (on die) or core 4 (via QPI)

Выделение памяти в NUMA системе

По умолчанию:
Страница памяти отображается на определённую планку при «дотрагивании» (записи), не при выделении памяти

taskset, numactl, likwid-pin



NUMA Operations

	cmd	option	arguments	description
Socket Affinity	numactl	-c	{0,1,2,3}	Only execute process on cores of this (these) socket(s).
Memory Policy	numactl	-l	{no argument}	Allocate on current socket.
Memory Policy	numactl	-i	{0,1,2,3}	Allocate round robin (interleave) on these sockets.
Memory Policy	numactl	--preferred=	{0,1,2,3} select only one	Allocate on this socket; fallback to any other if full .
Memory Policy	numactl	-m	{0,1,2,3}	Only allocate on this (these) socket(s).
Core Affinity	numactl	-C	{0,1,2,3, 4,5,6,7, 8,9,10,11, 12,13,14,15}	Only execute process on this (these) Core(s).

Лабораторная работа 3

Задача: реализовать примитив синхронизации "барьер" в системе с общей памятью.

Варианты задания

- глобальный "линейный" барьер
- глобальный "древовидный" барьер
- синхронизация с соседями (1D)

Тестирование проводить на задаче
"явная разностная схема для уравнения теплопроводности"
(исходный код с синхронизацией OpenMP #pragma omp barrier прилагается).

При необходимости использовать volatile переменные и атомарные операции.

Лабораторная работа 3

Во всех вариантах следует провести тестирование:

- при следующих размерах массива NxN:
  - Половина размера кэша L1 (~64x32),
  - Половина размера кэша L2 (~128x128),
  - Половина размера кэша L3 (~512x256),
  - В 10 раз больше размера кэша L3 (~2048x1024).

- переменные, используемые для синхронизации располагать в памяти тремя способами
  - плотная упаковка (int __h[128], обращение h[tid])
  - каждая переменная в отдельной кэш-строке (int h[128][64], обращение h[tid][0])
  - каждая переменная на отдельной странице памяти (int h[128][4096], обращение h[tid][0])

В качестве оценки производительности указывать время (минимальное и среднее) затраченное на синхронизацию на одной итерации.

Время работы барьера измерять с помощью rdtsc.