HPC Parallel Programming: Overview and Sequential Programming Optimization

Parallelization and Optimization Group
TATA Consultancy Services, SahyadriPark Pune, India

April 29, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   3.1 Sequential Programming Optimization.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   3.1 Sequential Programming Optimization.Today April 29, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization. Today May 2, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization. Today May 2, 2013
   - 3.4 Tools.

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization. Today May 2, 2013
   - 3.4 Tools. April 30 and May 2, 2013

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization. Today May 2, 2013
   - 3.4 Tools. April 30 and May 2, 2013
   - 3.5 Hands on training exercises. Afternoon

HPC Parallel Computing Course Overview

1. HPC Cluster Overview. Last week
2. Job Submission Cluster. Today: April 29, 2013
3. Parallel Programming:
   - 3.1 Sequential Programming Optimization. Today April 29, 2013
   - 3.2 Multicore Programming Optimization. Today April 30, 2013
   - 3.3 Multinode Programming Optimization. Today May 2, 2013
   - 3.4 Tools. April 30 and May 2, 2013
   - 3.5 Hands on training exercises. Afternoon
   - 3.6 Q&A.

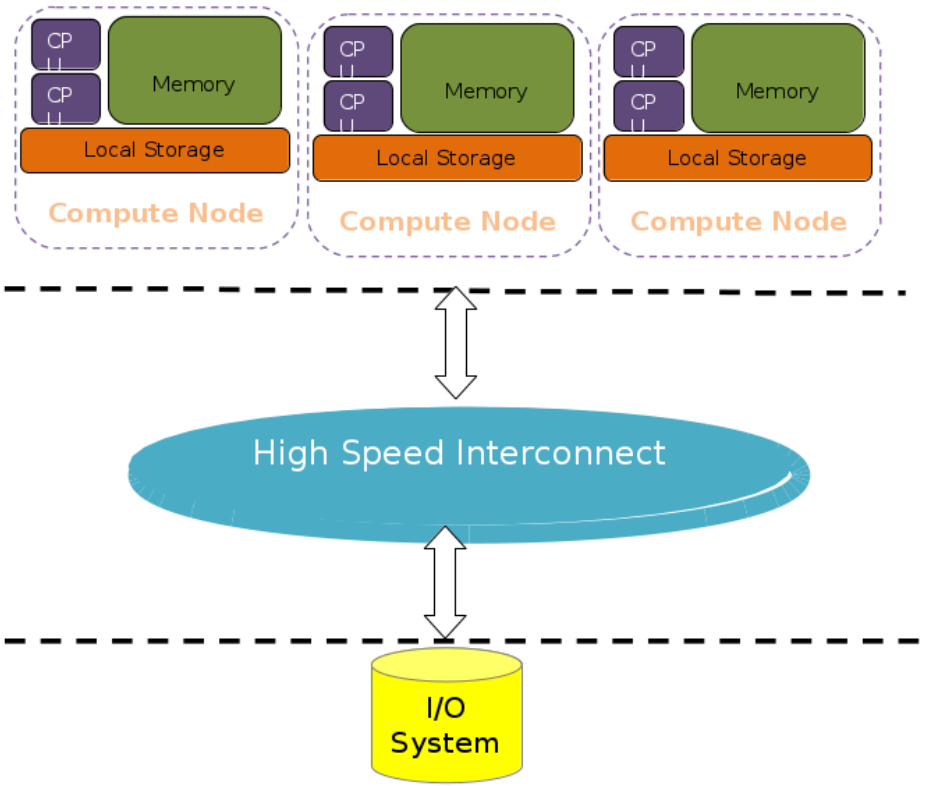
Acknowledgements

The Parallelization and Optimization group of the TCS HPC group have created and delivered this HPC training. The specific people who have contributed are:

1. OpenMP presentation and Cache/OpenMP assignments: Anubhav Jain, Pthreads presentation: Ravi Teja.
2. Tools presentation and Demo: Rihab, Himanshu, Ravi Teja and Amit Kalele.
3. MPI presentation: Amit Kalele and Shreyas.
4. Cache assignments: Mastan Shaik.
5. Computer and Cluster Architecture and Sequential Optimization using cache.Multicore Synchronization, Multinode Infiniband introduction and general coordination and overall review: Dhananjay Brahme.

HPC Computing Cluster:

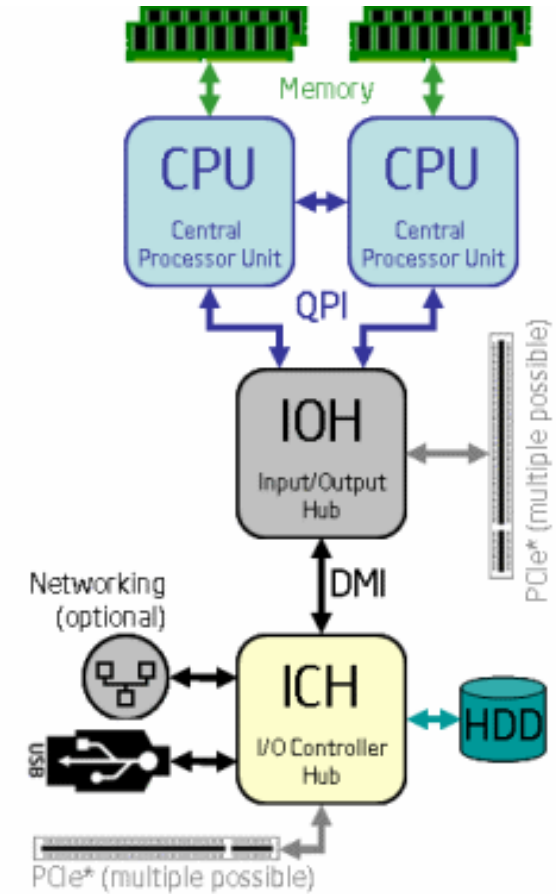
Figure: High Performance Multicore Multinode Cluster:



Source: Sanket Sinha, HPC Data Operations Presentation, TCS, Pune

Memory Access:

Figure: CPU to Memory connection
NUMA Source: www.intel.com



Memory Access:

Figure: CPU to Memory connection NUMA Source: www.intel.com

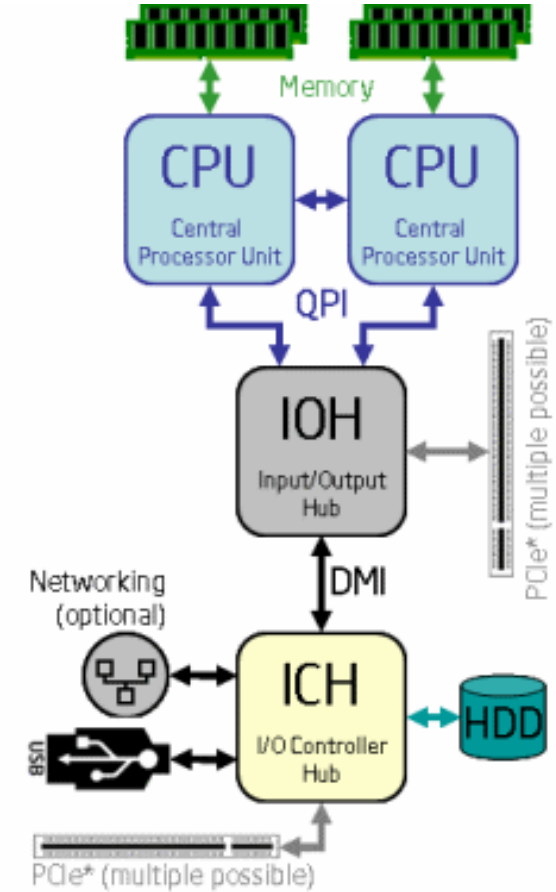
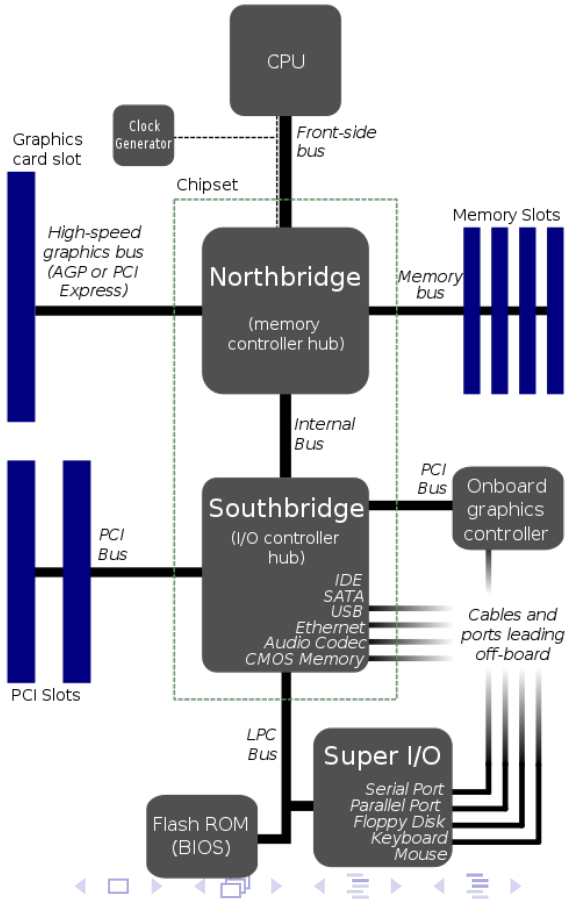
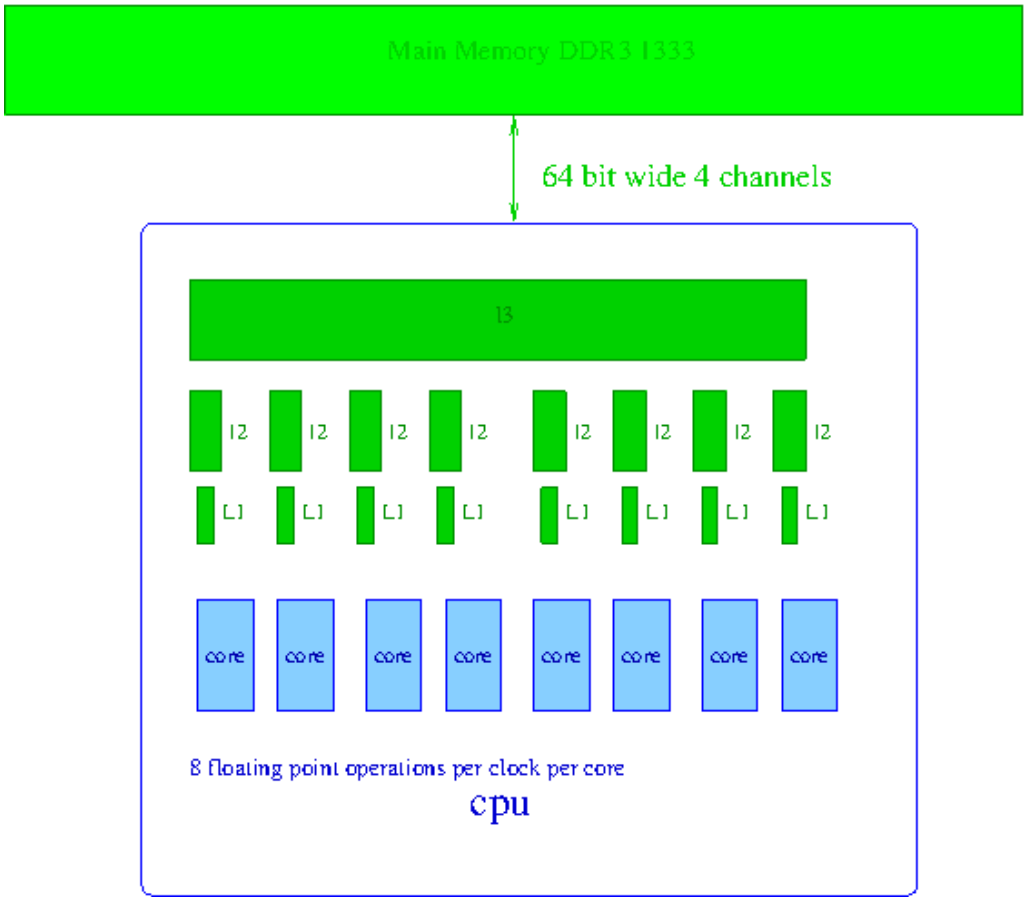


Figure: CPU to Memory connection via FrontSide Bus. Source: Wikipedia



CPU Memory Architecture

Figure: CPU cores, caches and Memory



CPU Memory Bandwidth: Sandy Bridge ES 2670

CPU Specs		Comment
No of Sockets	2	
Technology	32 nm	
No. of Cores	8	
Clock Rate	2.6 Ghz	
No. of Floating Point operations per clock	8	8*3*8 = 192 2.6 * 192= 499.2

CPU Memory Bandwidth: Sandy Bridge ES 2670

CPU Specs		Comment
No of Sockets	2	
Technology	32 nm	
No. of Cores	8	
Clock Rate	2.6 Ghz	
No. of Floating Point operations per clock per core	8	8*3*8 = 192 2.6 * 192= 499.2 499.2 * 8 = 3993.6 GB/s

CPU Memory Bandwidth: Sandy Bridge ES 2670

CPU Specs		Comment
No of Sockets	2	
Technology	32 nm	
No. of Cores	8	
Clock Rate	2.6 Ghz	
No. of Floating Point operations per clock per core	8	8*3*8 = 192 2.6 * 192= 499.2 499.2 * 8 = 3993.6 GB/s
QPI speed	8GT/s	
PCI Express 3	40 lane	

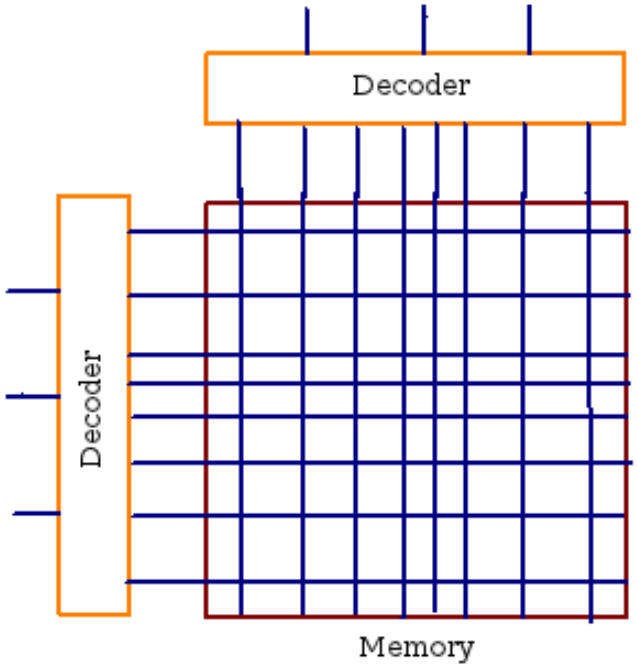
CPU Memory Bandwidth: Sandy Bridge ES 2670

CPU Specs		Comment
No of Sockets	2	
Technology	32 nm	
No. of Cores	8	
Clock Rate	2.6 Ghz	
No. of Floating Point operations per clock per core	8	8*3*8 = 192 2.6 * 192= 499.2 499.2 * 8 = 3993.6 GB/s
QPI speed	8GT/s	
PCI Express 3	40 lane	

Mem Specs		Comment
Memory Type	DDR3-800/ 1066/1333/ 1600	1333 * 8 bytes
No. of Channels	4	allows for parallel reads by the cpu
Memory CPU bus width	64 bits	
Max Memory Bandwidth	51.2GB/s	1333 * 8 * 4 = 42.656 GB/s
Max Memory Size	750 GB	

There is 100X gap between the CPU and Memory Bandwidth.

Solution: On Chip Memory



Solution: On Chip Memory

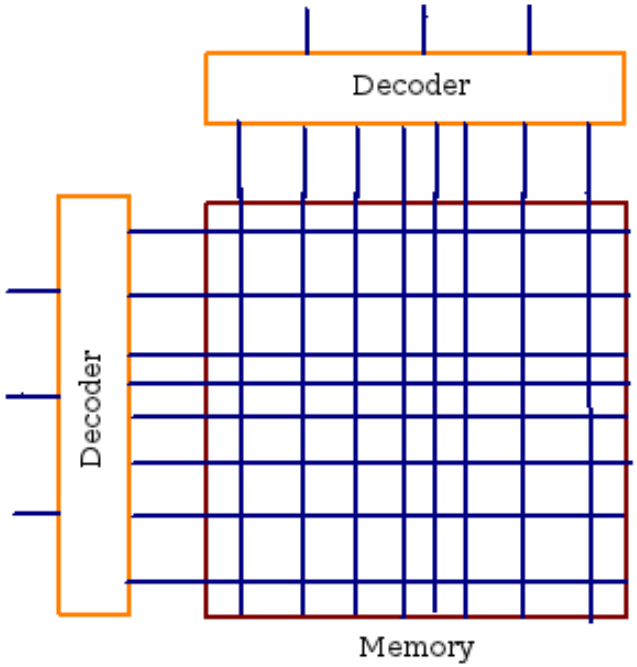
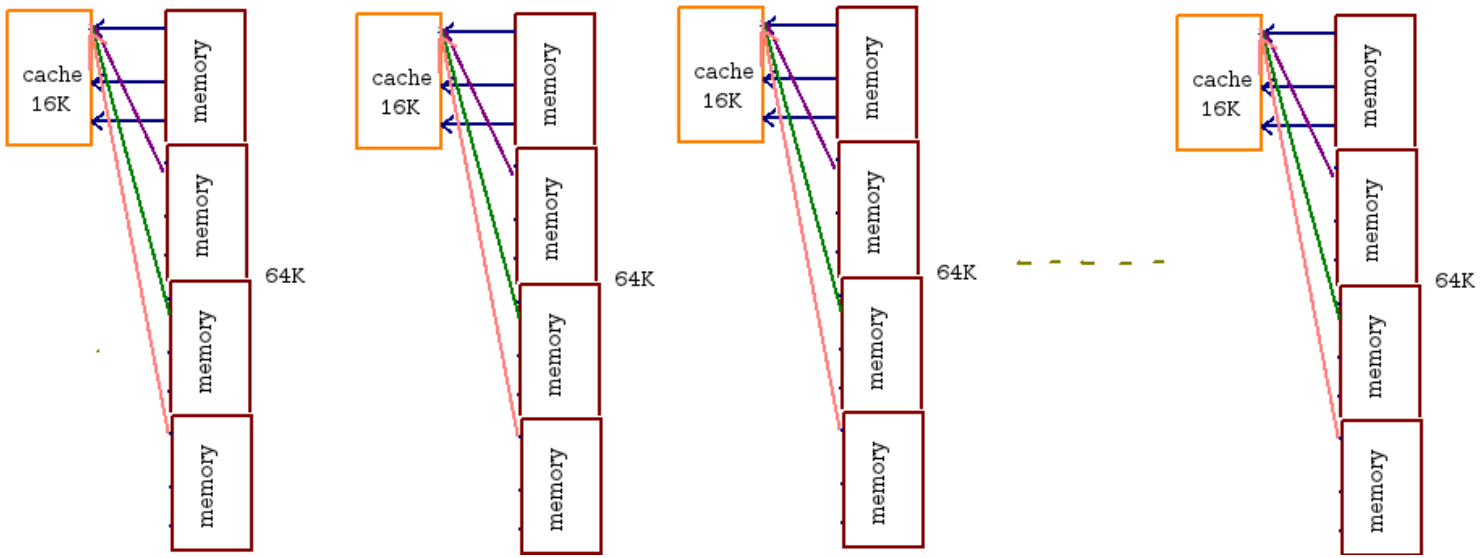


Table: Memory Hierarchy

	Cache1	**Cache2**	**Memory**	**Speed**
Size	32K	4Mb	2Gb	Decoding:Slower:$O(log(Size))$
Area	-	-	larger	Slower:$O(Size^{1/2})$
Speed	3 cycles	14 cycles	114 cycles	-
Technology	Static Ram	Static Ram	Dynamic Ram	Cheaper CMOS
Location	On-chip	On-chip	Of-chip	Slower:Larger Capacitance and Resistance

Cache Line

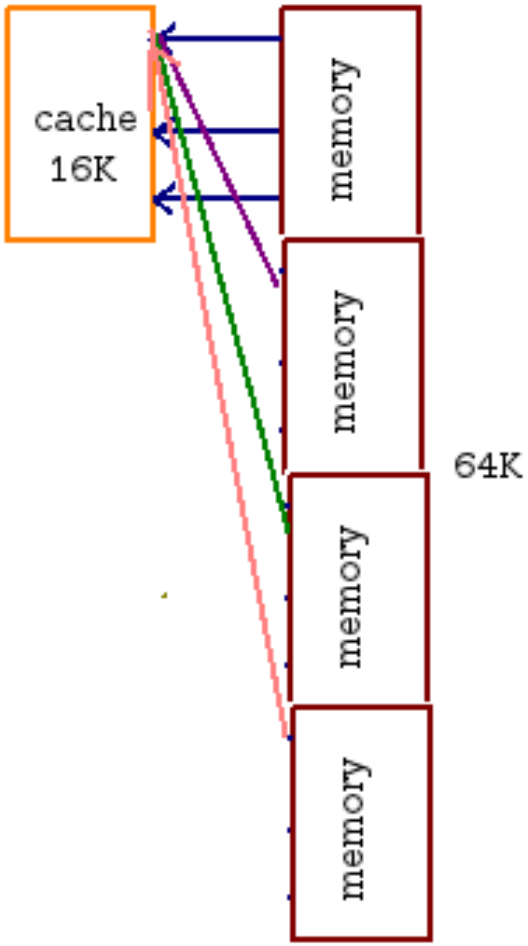
Figure: Cache Line is 4 (several) bytes



Cache Details

Topic	Policy
Cache Line Structure	Valid,Address Bits
Write Policy	Write Back or Write Thru
Cache Line replacement	Least recently used

Direct Mapped Cache



Principle	**Implication**
Resolve Mapping	Store higher address with data
Resolve Mapping	Compare the higher address
Locality	Lower bits map directly higher bits cause overlap
Overlap?	Problem

Set Associative Cache

Figure: With cache size doubled, overlap is reduced by 2

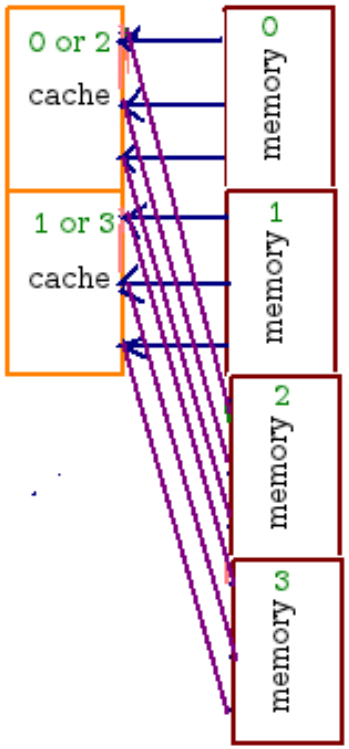
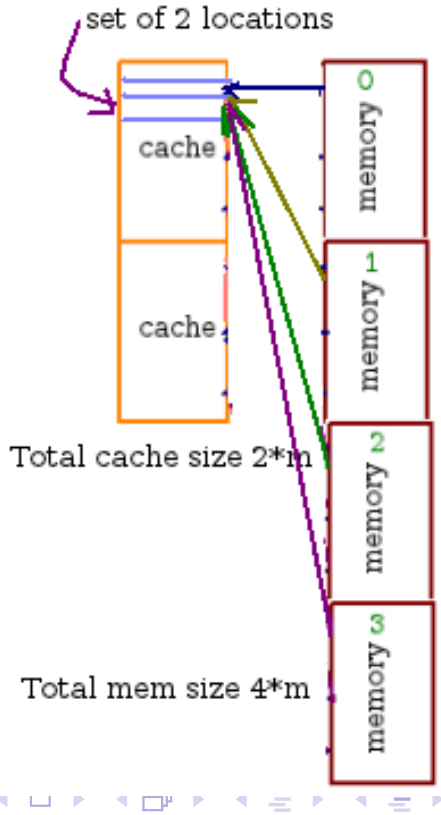


Figure: With cache size doubled, data from any 2 out of 4 regions is stored



Set Associative (Contd):

Problem	**Problem**
Direct Mapped	Choice Restricted to 1 out of 2 memory regions.
Set Associative	Allow $^{2*2}C_2$ for each of the m sets in the cache

Programming

Programming methodology to use cache efficiently

1. Principle: Use a cache line in as many computations as possible. This reduces Cache misses.
2. Method:
   - 2.1 loop blocking.
   - 2.2 nested loop: interchange loops.
3. Application:
   - 3.1 Array access: Access array consecutively: Consider an array of 1M doubles. Initialize each element to 1.5 and compute the sum by adding up each consecutive element. How long did it take? Compute the sum by adding up each 11^{th} element till you have added all the elements. How long did it take?
   - 3.2 Matrix Transpose: block transpose.
   - 3.3 MatrixXMatrix:interchange loops, block on loop.

More optimization

1. Reduce computation:
2. Application:
   2.1 Remove loop invariant outsize.
   2.2 Loop unrolling.
3. Replace expensive operation by cheaper operation:
4. Application:
   4.1 Multiplication by power of 2 by shift

Assignments

1. Write a program to transpose matrix of 8192 X 8192 doubles in the normal way. Now implement a version that is optimized for cache. Assume a cache line has 64 bytes, i.e., 8 doubles.
2. Write a program to multiply two matrices of 2048 X 2048 doubles in the normal way. Improve the efficiency by reordering inner two loops. Compute B^T and use this matrix to compute A X B. How long did it take? Use blocking and compute A X B. How long did it take?

Thank You