CHALMERS TEKNISKA HÖGSKOLA
Institutionen för data- och informationsteknik
Avdelning för datorteknik

Re-sit exam in EDA284 (Chalmers) and DIT361 (GU) Parallel Computer Architecture, Monday, January 7th, 2019, 14:00h - 18:00h

---

Teacher/Lärare: Miquel Pericàs, tel 7721705

Language/Språk: Answers shall be given in English.

Solutions/Lösningar: Solutions will be posted on Wednesday, January 9th, on the course's pingpong page.

Exam review/Granskning: The review date will be posted on the course pingpong page by the time you receive the email from LADOK.

Grades:

Chalmers				
Points	0-23	24-35	36-47	48-60
Grade	Failed	3	4	5

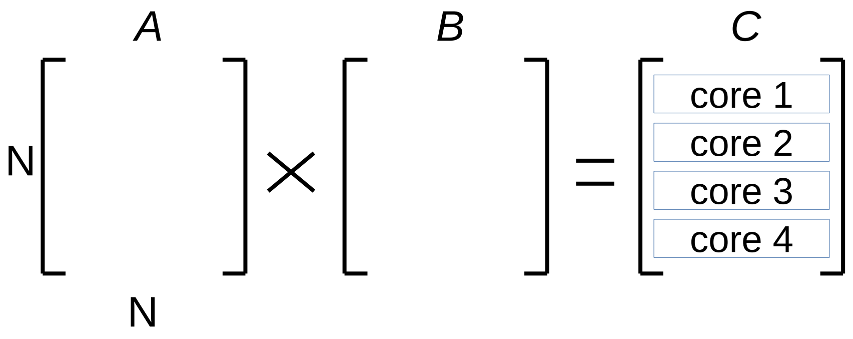
GU				
Points	0-23	24-41	42-60	
Grade	Failed	G	VG	

Good Luck!

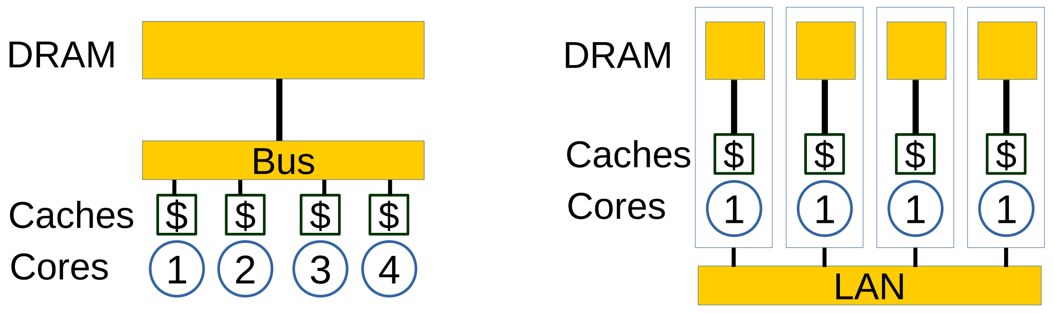
Problem 1 (10 points)

The two main programming paradigms for parallel computers are shared memory and message passing. In the course, the parallelization of a matrix multiplication ($A \cdot B = C$) was used to exemplify both paradigms. In this problem you will compare the performance of both approaches.

Assume that all matrices are square and each consists of N rows and columns. A typical way to decompose (i.e. parallelize) the problem over four cores is shown in the following figure:



This strategy can be used to parallelize the algorithm for both (a) shared memory and (b) message passing systems. The figure below shows two such systems.



(a) shared memory (b) message passing

The task is to discuss whether the five following statements are correct in the context of the two paradigms and the two shown systems. You should write 1-2 sentences for each statement and paradigm (i.e., there should be 10 answers). Just stating *true* or *false* will not be considered sufficient.

STATEMENT A (ST. A): “In order to function correctly, it is necessary to explicitly copy both matrices A and B into the DRAM memory connected to each core.”

ST. B: “As long as the Bus /LAN interconnect bandwidth is larger than 16 GB/s, the execution time is independent of the speed of the Bus / LAN interconnect.”

ST. C: “The execution time does not depend on the DRAM bandwidth, as long as the DRAM bandwidth exceeds 8 GB/s.”

ST. D: “As the matrix size N increases, the speed-up compared to a single core approaches 4.0 .”

ST. E: “Neither of the two parallelization paradigms and requires Operating System support.”

Unless otherwise specified, the following assumptions are to be considered:

- Initially the matrices *A* and *B* are stored in the DRAM memory connected to core 1. The algorithm finalizes when matrix *C* is stored back in the DRAM memory of core 1.
- In both systems the DRAM, Bus and LAN support up to 32 GB/s of bandwidth.
- The matrix multiplication is parallelized into multiple threads, each consuming a constant 4 GB/s of DRAM bandwidth on each core.
- The DRAM capacity is never a limiting factor.
- The cache coherence protocol used in the shared memory system (a) is MSI-invalidate.

Problem 2 (5p)

Consider a shared memory multiprocessor that consists of four processor/cache units and where cache coherence is maintained by an MSI-invalidate protocol. The table shows the access sequence taken by the four processors to the same block but to different variables (A,B,C,D) in that block.

Nr	**Processor 1**	**Processor 2**	**Processor 3**	**Processor 4**
1	RA			
2		RB		
3			RC	
4				RD
5	WA			
6		WB		
7			WC	
8				WD
9				RD
10			RD	
11		WB		
12	RA			

Your task is to:

(a) Classify each access as a hit, cold miss, true sharing miss, or false sharing miss. Furthermore, which of the misses could be ignored and still guarantee that the execution is correct?

(b) If each bus-upgrade costs 10 bytes, and a bus read request costs 38 bytes (32 for cache block + 6 bytes bus-header) what is the total essential traffic of these accesses?

Problem 3 (12p)

A design team needs to choose an appropriate cache coherence protocol to be used for a shared memory multiprocessor with a number of processor/private cache units connected by a shared single-transaction bus. The team is considering three invalidation, snooping-based protocols: MSI, MESI and MOESI.

In order to select the protocol, the team is analyzing the following representative sequence of accesses happening on the bus in the following order:

R1/X, R2/X, R2/Y, W1/X, W2/Y

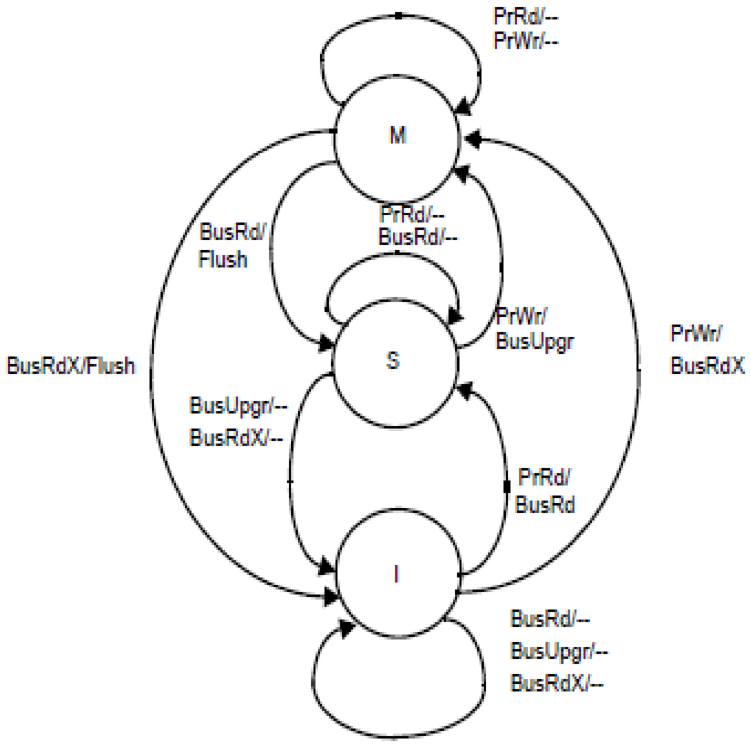
where the notation R*i*/X, W*i*/X means a read and write from processor *i* to cache block X. The latency and traffic costs of different operations considering a block size of B bytes are:

Operation	**Latency**	**Bus Traffic**
Read hit	1	N/A
Write hit	1	N/A
Request serviced by next level (BusRd,Flush)	100	6+B
Request serviced by different cache (BusRd/Flush)	20	6+B
Bus upgrade (BusUpgr)	10	10
Snoop action	5	N/A

Assumptions:

1. A bus upgrade consists of transferring the request on the bus and making a snoop action in each cache (the latency of the latter is shown in the table)
2. There is no contention on the tag directory (assume duplicated tag directory)
3. X and Y are not the same cache block
4. The block size B is 32 bytes
5. Read-exclusive requests (BusRdX) have the same latency and traffic as Read requests

The MSI state-transition diagram is shown below



To assess the performance of the three protocols the team must construct a table with the latency (cycles) and bus traffic (bytes) needed to complete the aforementioned access sequence (shown below).

Protocol	Latency (cycles)	Traffic (bytes)
MSI		
MESI		
MOESI		

Your task is to:

(a) Describe in 1-2 sentences the meaning of the states ‘E’ and ‘O’ present in the MESI and MOESI protocols. What problem do they address?

(b) Show the protocol transitions for the given sequence in both caches and for all three protocols.

(c) Calculate and complete the above table with the latency and traffic values

Problem 4 (8p)

Consider a future 8-way CMP on which you can power off some cores to allow the rest to operate at a higher frequency. You can use either 1, 2, 4, or 8 cores at the respective frequencies:

- when only one core is running it operates at 0.3ns clock cycle
- when two cores are running they operate at 0.4ns clock cycle
- when 4 cores are running they operate at 0.5ns clock cycle
- when all 8 cores are running they operate at 1ns clock cycle

Consider a partially parallel application which has a serial part that is 1000 Instructions and a parallel part that can be parallelized at will which is 2000 Instructions. Parallelizing however requires you to create threads in the serial part of the application and 100 Instructions are added for every thread you create. The serial and parallel part of the applications all run one instruction per cycle regardless of the frequency.

<u>Your task is to:</u>

(a) Calculate the execution time of the application for the above processor when using 1,2,4,8 cores at their respective frequency without reconfiguring the processor while the application is running. Which configuration is better?

(b) What if you could reconfigure the processor to run the serial part with one core at 0.3 ns clock cycle and then configure it to 2 or 4 or 8 cores for the parallel part. What is the execution time for each case? Always consider that creating each thread takes 100 Instructions that are added to the serial part (and run on the single core at 0.3ns clock cycle).

Problem 5 (8p)

Two common spinlocks relying on atomic synchronization primitives are the Test-and-Set-based spinlock and the Test-and-Test-and-Set spinlock. The code for the two spinlocks is shown below.

Action/Label	**`Lock #1 (Test-and-Set)`**	**`Lock #2 (Test-and-test-and-set)`**

`Lock:`	`T&S R1, _lock`
	`BNEZ R1, Lock`
		`LW R1, _lock`
		`BNEZ R1, Lock`
		`T&S R1, _lock`
		`BNEZ R1, Lock`

`Unlock:`	`SW R0, _lock`	`SW R0, _lock`

Assume an MSI-invalidate snoopy cache coherent protocol is used to keep the caches of a shared memory multiprocessor consisting of four processors coherent. In the above code, assume that R0 = 0 and that `_lock` is the variable holding the lock.

In the execution under consideration, all four threads are trying to modify a shared data structure protected via a single global lock. When one thread obtains access to a lock, it keeps the lock for a long period of time before releasing it.

<u>Your task is to:</u>

(a) Describe the sequence of MSI protocol transitions that will occur for both **Lock #1** and **Lock #2**.

(b) Explain which spinlock behaves better in this scenario and why?

(c) If the cache coherence protocol were MESI or MOESI, are there any differences in the behavior?

(d) Assume now that the different threads usually modify different parts of the shared data structure. Under these circumstances, a single global lock is an inefficient solution to the locking problem. Can you think of any better solutions? Please list at most two alternative solutions.

Problem 6 (6p)

A system architect has three choices for an on-chip interconnection network: a uni-directional ring (UR), a bi-directional ring (BR) and an NxN mesh network (NM). In addition, the architect has two choices for providing coherence between L1 caches: snoop-based (SB) or directory-based (DB). There are 16 cores on the chip. All cores have private L1 caches and they share a L2 cache that is divided into 16 banks, with one bank attached to each core. The latency to communicate between two cores connected directly by a link is 1 cycle. For directory-based coherence the access time to the directory is 5 cycles. Assume that there is no contention in any link or in the access to the directory.

(a) Which combination of design choices provides the shortest latency to provide coherence? In this context, latency is considered to be the time it takes to reach all potential destination cores (not including acknowledgments). Consider the worst and average case for a coherence message to reach its destination(s)

(b) Now consider 256 cores and a directory access time of 15 cycles, which combination of design choices provides the shortest latency to provide coherence in this case?

Please organize the results in a table as follows:

Case (a) 16 cores

Latency	**Coherence Type**	**UR**	**BR**	**NM**
Worst Case	SB			
Worst Case	DB			
Average	SB			
Average	DB			

Case (b) 256 cores

Latency	**Coherence Type**	**UR**	**BR**	**NM**
Worst Case	SB			
Worst Case	DB			
Average	SB			
Average	DB			

Problem 7 (5p)

In order to implement a multiprocessor system that behaves correctly, it is necessary to keep caches coherent and to implement a precise memory consistency model.

Your task is to describe:

(a) What is the difference between coherence and consistency? (1-2 sentences)

(b) Briefly describe the two memory consistency models known as Sequential consistency (SC) and Relaxed Memory Order (RMO)? What relaxations do they allow?

(c) Which of these two models (SC and RMO) is more restrictive in terms of hardware implementation? How do these two models handle store buffers?

Problem 8 (6p)

Core multithreading is a technique to improve the utilization of resources in modern processors whose performance is limited by long latency events such as cache misses. The goal of this problem is to compare the differences between the four models of multithreaded processors that have been introduced in class: (1) block multithreading, (2) interleaved multithreading, (3) barrel processors, and (4) simultaneous multithreading.

<u>For each of the four alternatives, your task is to briefly describe (1-2 sentences):</u>

(a) The granularity of thread switches, i.e. when are threads switched?

(b) In which types of pipeline (e.g. in-order, out-of-order) can the technique be implemented? What is the cost of a thread switch in each case?

(c) What should the software (i.e. the running applications, not the operating system) provide to achieve good resource utilization?