Advanced and parallel architectures

Prof. A. Massini

June 1, 2017

Student Name

_____

Matricola

_____

Exercise 1a (3 points)	
Exercise 1b (3 points)	
Exercise 2 (3 points)	
Exercise 3 (3 points)	
Exercise 4a (8 points)	
Exercise 5 (4 points)	
Exercise 6 (3 points)	
Exercise 7 (2 points)	
Exercise 8 (3 points)	
Total (32 points)	

Exercise 1a (3 points) – Interconnection Networks – CLOS

Design a Clos network of size 128 x 128, using in the first stage modules having 16 inputs. Consider both cases, strictly non-blocking and rearrangeable network.

Exercise 1b (3 points) – Interconnection Networks – Comparison Clos-Crossbar

Compare the cost of the crossbar 128 x 128 and the Clos network strictly non-blocking and rearrangeable designed in the previous point.

Exercise 2 (3 points) – Interconnection networks

Complete the scheme of the Butterfly and Baseline networks of size N=8.

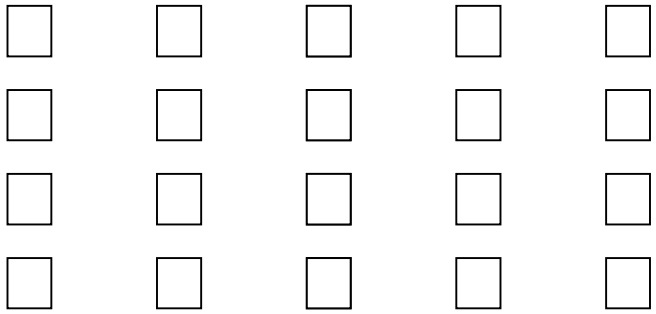
Write the permutation obtained by rotating left the binary representation of inputs (in the case of 3 bit): $P = \begin{pmatrix} 01\,23\,45\,67 \\ \end{pmatrix}$

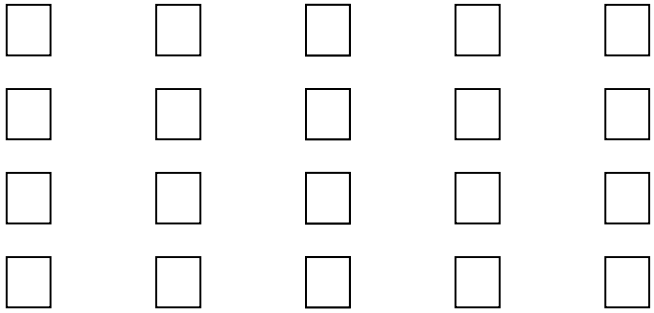
Show if the Butterfly and Baseline Networks can realize this permutation, by showing the switch setting, using the self-routing.

Exercise 3 (3 points) – Interconnection networks

Complete the scheme of the Baseline-Baseline^{-1} and on a Butterfly-Butterfly^{-1}.

Show the switch setting to realize permutation $P = \begin{pmatrix} 01234567 \\ 17506243 \end{pmatrix}$ on a Baseline-Baseline^{-1} and on a Butterfly-Butterfly^{-1} according to the looping algorithm.





Exercise 4 (3+3+2 points) - GPU & CUDA

You need to write a kernel that operates on a 3D matrix of size **700x700x500**. You would like to assign one thread to each matrix element. You would like your thread blocks to use the maximum number of threads per block possible on your device, having compute capability 1.3, assuming **three-dimensional** blocks.

a) How would you select the grid dimensions and block dimensions of your kernel?

b) How would you select the grid dimensions and block dimensions of your kernel, if you assume blocks are cubic (that is the three dimensions have the same size)?

c) How many idle threads do you expect to have, in both cases?

Technical specifications	Compute capability (version)									
	1.0	**1.1**	**1.2**	**1.3**	**2.x**	**3.0**	**3.5**	**3.7**	**5.0**	**5.2**
Maximum dimensionality of grid of thread blocks	2				3					
Maximum x-dimension of a grid of thread blocks	65535					2^{31}-1				
Maximum y-, or z-dimension of a grid of thread blocks	65535									
Maximum dimensionality of thread block	3									
Maximum x- or y-dimension of a block	512				1024					
Maximum z-dimension of a block	64									
Maximum number of threads per block	512				1024					
Warp size	32									
Maximum number of resident blocks per multiprocessor	8					16			32	
Maximum number of resident warps per multiprocessor	24		32		48	64				
Maximum number of resident threads per multiprocessor	768		1024		1536	2048				
Technical specifications	**1.0**	**1.1**	**1.2**	**1.3**	**2.x**	**3.0**	**3.5**	**3.7**	**5.0**	**5.2**
	Compute capability (version)									

Exercise 5 (4 points) - Cache coherence

Consider a multicore multiprocessor implemented as a symmetric shared-memory architecture, as illustrated in the figure.

Each processor has a single, private cache with coherence maintained using the snooping coherence protocol. Each cache is direct-mapped, with four blocks each holding two words. The coherence states are denoted M, S, and I (Modified, Shared, and Invalid).

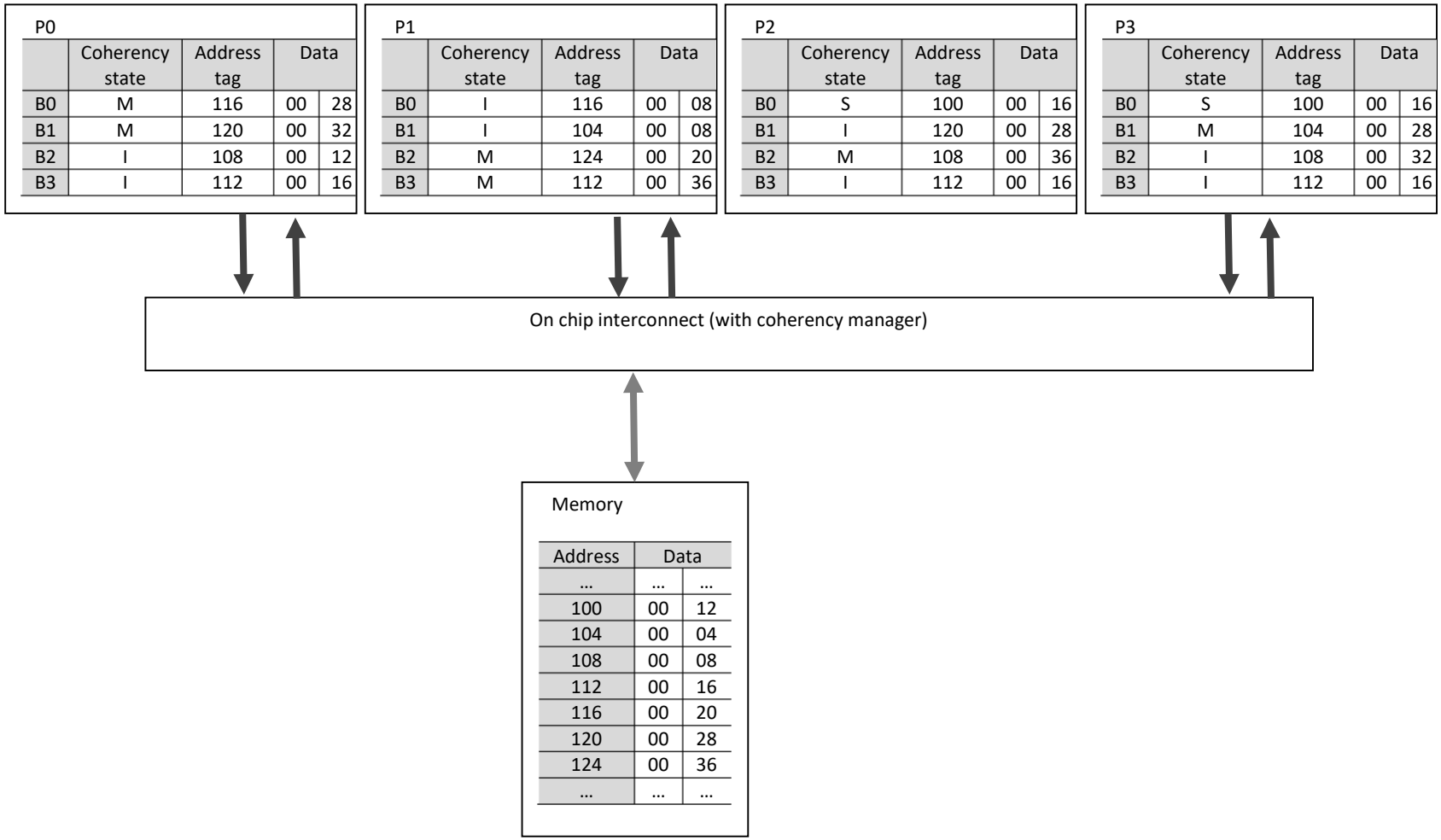
Each part of this exercise specifies a sequence of one or more CPU operations of the form:

P#: <op> <address> [<value>]

where P# designates the CPU (e.g., P0), <op> is the CPU operation (e.g., read or write), <address> denotes the memory address, and <value> indicates the new word to be assigned on a write operation.

For each part of this exercise, assume the initial cache and memory state as illustrated in the figure.

Show in the table for the results:
- miss/hit
- the coherence state before the action
- the CPU processor Pi and cache block Bj
- the changed state (i.e., coherence state, tags, and data) of the caches and memory after the given action.



P0	Coherency state	Address tag	Data	
B0	M	116	00	28
B1	M	120	00	32
B2	I	108	00	12
B3	I	112	00	16

P1	Coherency state	Address tag	Data	
B0	I	116	00	08
B1	I	104	00	08
B2	M	124	00	20
B3	M	112	00	36

P2	Coherency state	Address tag	Data	
B0	S	100	00	16
B1	I	120	00	28
B2	M	108	00	36
B3	I	112	00	16

P3	Coherency state	Address tag	Data	
B0	S	100	00	16
B1	M	104	00	28
B2	I	108	00	32
B3	I	112	00	16

Address	Data	
...	...	...
100	00	12
104	00	04
108	00	08
112	00	16
116	00	20
120	00	28
124	00	36
...	...	...

Specify the value returned by a read operation.

a) P2: write 104 ← 12

Comments

hit/miss	state before	Pi.Bj (state, tag, datawords)

b) P0: read 108

hit/miss	state before	Pi.Bj (state, tag, datawords)

c) P3: write 120 ← 24

hit/miss	state before	Pi.Bj (state, tag, datawords)

d) P2: write 116 ← 20

hit/miss	state before	Pi.Bj (state, tag, datawords)

Exercises 6 (3 points) Performance equation & Amdhal Law

Three enhancements with the following speedups are proposed for a new architecture:
$Speedup_1 = 10$
$Speedup_2 = 25$
$Speedup_3 = 20$
Assume for some benchmark, the fraction of use is 50% for each of enhancements 1, 25% for enhancement 2 and 35% for enhancement 3.
What enhancement should be implemented to maximize performance?

Exercises 7 (2 points) Amdhal Law

The following measurements are recorded with respect to the different instruction classes for the instruction set running a given set of benchmark programs:

Instruction Type	Instruction Count (millions)	Cycles per Instruction
Arithmetic and logic	6	1
Load and store	4	3
Branch	1	4
Others	5	3

Assume that "*branch*" instructions can be modified so that they take 2 cycle per instruction instead of 4, and "load and store" instructions can be modified so that they take 2 cycle per instruction instead of 3 as in the table. Compute the speed up obtained by introducing these two enhancement using the Amdhal law.

Exercises 8 (3 points) Performance equation

Suppose we have made the following measurements, where we are considering FP (Floating Point) instructions and FPD (Floating Point Division) instructions:

Frequency of FP operations = 30%
Average CPI of FP operations = 3.0
Average CPI of other instructions = 1.5
Frequency of FPD = 5%
CPI of FPD = 10

Assume that the two design alternatives are to decrease the CPI of FPD to 2 or to decrease the average CPI of all FP operations to 2.0.
Compare these two design alternatives using the processor performance equation.