Department of informatics
Computer Science degree
Computer Architecture

Universidad Carlos III de Madrid

Final exam

June 26th 2014

Rules:

- Read the exam carefully before starting
- Mobile phones, electronic devices, books and notes are not allowed.
- Mobile phones have to be completely disconnected
- Write the exam using a pen (not a pencil)
- **Exam duration is 180 minutes**
- **Use a different page for each exercise. If you don't answer one question, use a white page with the exercise number and your name.**

NAME:
SURNAMES:
NIA / ID:

Exercise 1: (15 points) Indicate the objective of the following cache optimization techniques performed by the compiler:

- Reorder procedures
- Align the basic blocks to cache blocks
- Place close basic blocks if a branch is highly probable taken

NOTE: It is not necessary to describe the optimization technique but only its objective.

SOLUTION

- **Reorder procedures: reduce the conflict cache misses that are produced when two procedures are associated to the same cache lines.**
- **Align the basic blocks to cache blocks: reduce the number of cache misses for sequential codes (less cache blocks are read).**
- **Place close basic blocks if a branch is highly probable taken: reduce the number of cache misses related to conditional branches.**

Exercise 2: (15 points) Describe the following ordering rules of the C++ programing language:

- Synchronizes-with: relationship for atomic types.
- Happens-before

SOLUTION:

- **Synchronizes-with: relationship for atomic types. Write-read synchronizations. Sequence of read-modify-write operations or Written by the write operation. Or Written by a next write belonging to the same thread that performed the write.**

Final exam

June 26th 2014

- **Happens-before: Specifies the operation that "sees" the result of another operation. The sentences are ordered for the same thread (There is not ordering for the operations of the same sentence) and for different threads, there is a "before" order when (There is a synchronization relationship or There are "happens-before" conditions).**

Exercise 3: (15 points): Two threads execute the following instructions in a computer based on the Intel architecture. There are two variables (z and t) with the initial value of 42.

Thread 1	Thread 2

`mov [_z], 1`
`mov r1, [_z]`
`mov r2, [_t]`
	`mov [_t], 1`
	`mov r3, [_t]`
	`mov r4, [_z]`

Is it possible that after executing the Thread 2, r2 and r4 registers have the value of 42? Justify your answer.

SOLUTION

It is possible because the writes can be seen in a different order by each processor. In this way, thread 1 can have r1=1, r2=42 and thread 2 can have r3=1, r4=42.

Universidad Carlos III de Madrid

Department of informatics
Computer Science degree
Computer Architecture

Final exam

June 26th 2014

Exercise 4: (30 points) A given processor has associated the following latency table:

Instruction producing result	Instruction using result	Latency
ALU FP operation	Other ALU FP operation	6
ALU FP operation	Store double	3
Load double	ALU FP operation	2
Load double	Store double	0

The following code is executed in this processor:

```
LOOP:
L.D F0, 0(R1)
L.D F2, 0(R2)
ADD.D F4, F0, F2
S.D F4, 0(R3)
DADDUI R1, R1, #-8
BNE R1, R4, LOOP
```

The initial values of the registers are the following ones:

- R1: Last element of the first source array
- R2: Last element of the second source array
- R3: Last element of the destination array
- R4: Predefined: 8(R4) is the first element of the first source array.

All the arrays have 4000 entries.

Complete the following tasks:

1. Obtain the number of cycles necessary to execute all the iterations of the original code.
2. Obtain the number of cycles necessary to execute all the iterations of the code when you rewrite code to minimize stalls.
3. Obtain the number of cycles necessary to execute all the iterations of the code when you unroll the loop two times.
4. Obtain the number of cycles necessary to execute all the iterations of the code when you unroll the loop four times.

Final exam

June 26th 2014

SOLUTION

The original loop will be

L.D F0, 0(R1)
L.D F2, 0(R2)
<stall> x 2
ADD.D F4, F0, F2
<stall> x 3
S.D F4, 0(R3)
DADDUI R1, R1, #-8
BNE R1, R4, BUCLE

Each iteration requires of 11 cycles so the complete loop takes 44,000 cycles.

Section 2:

DADDUI can be moved.

L.D F0, 0(R1)
L.D F2, 0(R2)
DADDUI R1, R1, #-8
<stall> x 1
ADD.D F4, F0, F2
<stall> x 3
S.D F4, 0(R3)
BNE R1, R4, BUCLE

Each iteration requires of 10 cycles so the complete loop takes 40,000 cycles.

Section 3:

L.D F0, 0(R1)
L.D F2, 0(R2)
L.D F6, -8(R1)
L.D F8, -8(R2)
DADDUI R1, R1, #-16
ADD.D F4, F0, F2
ADD.D F10, F6, F8
<stall> x 2
S.D F4, 0(R3)
S.D F10, -8(R3)
BNE R1, R4, BUCLE

Final exam

June 26th 2014

Each iteration requires of 12 cycles so the complete loop takes 24,000 cycles.

Section 4:

```
L.D F0, 0(R1)
L.D F2, 0(R2)
L.D F6, -8(R1)
L.D F8, -8(R2)
L.D F12, -16(R1)
L.D F14, -16(R2)
L.D F18, -24(R1)
L.D F20, -24(R2)
DADDUI R1, R1, #-32
ADD.D F4, F0, F2
ADD.D F10, F6, F8
ADD.D F16, F10, F12
ADD.D F22, F18, F20
S.D F4, 0(R3)
S.D F10, -8(R3)
S.D F16, -16(R3)
S.D F22, -24(R3)
BNE R1, R4, BUCLE
```

Each iteration requires of 18 cycles so the complete loop takes 18,000 cycles.

Exercise 4: (25 points): Given a symmetric-memory multiprocessor based on the snooping bus protocol. Each processor has a private cache which is based on the MSI coherence protocol. Each cache line consists of a single word.

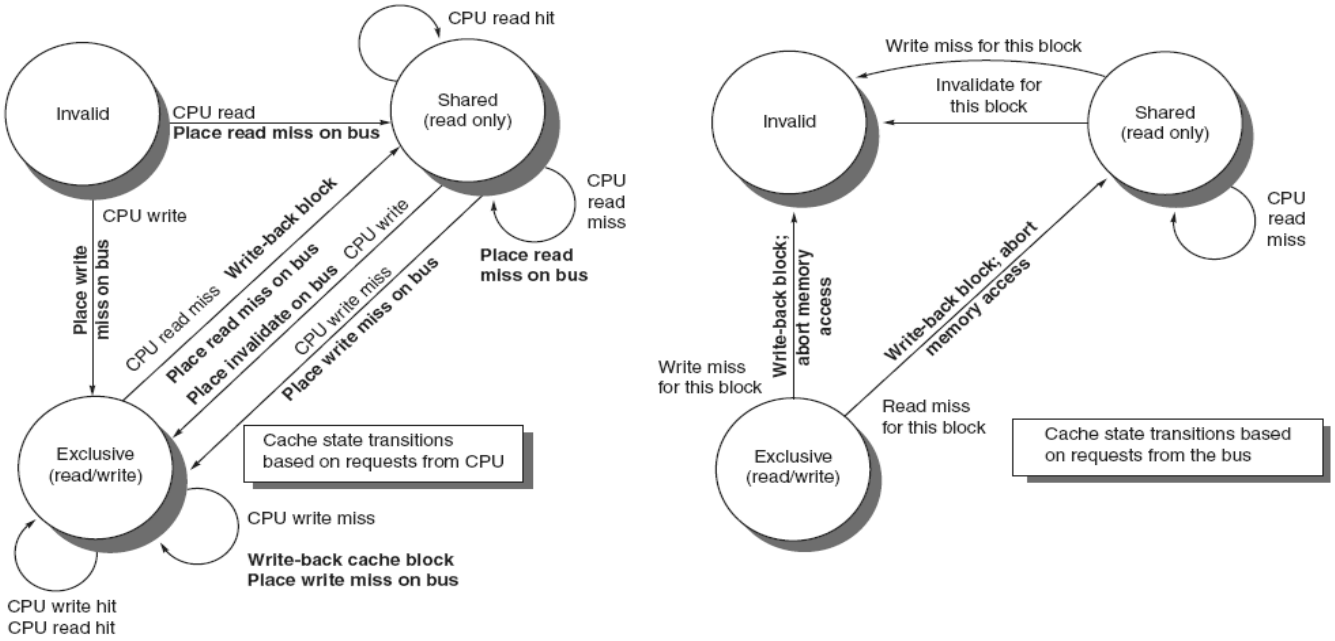
Department of informatics
Computer Science degree
Computer Architecture

Universidad Carlos III de Madrid

ARCOS

Final exam

June 26th 2014



The following table shows the initial value in each cache of four different variables.

	Initial state			
Processor	**A**	**B**	**C**	**D**
P0	*Shared*	*Exclusive*	*Shared*	*Shared*
P1	*Invalid*	*Invalid*	*Invalid*	*Shared*
P2	*Invalid*	*Invalid*	*Shared*	*Shared*

The following table shows the final state of these variables after performing several memory accesses.

	Final state			
Processor	**A**	**B**	**C**	**D**
P0	*Invalid*	*Invalid*	*Invalid*	*Shared*
P1	*Invalid*	*Invalid*	*Shared*	*Exclusive*
P2	*Exclusive*	*Exclusive*	*Invalid*	*Shared*

Universidad
Carlos III de Madrid

Department of informatics
Computer Science degree
Computer Architecture

Final exam

June 26th 2014

Complete the following tasks:

- Describe for each variable (A, B, C and D) what memory access/accesses are performed to reach the final state. Note 1: To reach the final state it could be necessary to perform a single or several memory accesses. Note 2: it is possible to have an unreachable final state (that is, a state without solution). Justify your answer.
- For each variable describe the generated bus traffic associated to the transition from the initial to the final state.

SOLUCIÓN:

- **A**

  P2 Writes A (exclusive) and invalidates the copy on P0

  P2 generates a write misss that is captured by P0. No traffic is generated.

- **B**

  P2 Writes B (estado a exclusive) and invalidates the copy on P0

  P2 generates a write misss that is captured by P0 . P0 writes-back the cache block which is sent to P2.

- **C**

  P1 writes B (estado a exclusive) and invalidates all the remaining copies.

  P1 generates a write misss that is captured by P0 and P2. No traffic is generated.

  P1 reads o writes D (another variable that has associated the same cache line as C). This produces a conclict miss and C block is replaced. Given that C was modified it is written in the main memory (bus traffic) and C is not present in p1 cache. Then, P1 reads C, generating a read miss and transiting to the shared state.

- **D**

  The final state is unreachable because a given block is no allowed to be exclusive and shared at the same time.

Universidad
Carlos III de Madrid

Department of informatics
Computer Science degree
Computer Architecture

Final exam

June 26th 2014