Computer Architecture and Engineering - 3rd year Bachelor in Informatics - ETSINF - UPV

Exercises of Unit 1. “Introduction to computer architecture”

Exercise 1.1.

Two alternative architecture designs for a computer (let’s name them A and B) are available. Both affect the clock frequency and the number of *l*oad/store instructions in programs. Taking into account that:

- The clock frequency of A is 5% faster than the one of B;
- A program compiled for A has 30% of *load/store* instructions, while the same program compiled for B reduces in 1/3 the number of such type of instructions;
- CPI is 1 for all instructions;

Decide which architecture design is better.

Exercise 1.2.

In a *Load/Store* computer, it has been measured the frequency of occurrence, and the CPI, of the following types of instructions:

Operation	%	CPI
ALU	43	1
LOAD	21	2
STORE	12	2
Branches	24	2

It has also been observed that 25 % of ALU instructions have a source operand that is never reused. Thus, it is proposed to enrich the instruction set of the computer with new arithmetic instructions having one source operand in memory, as it is shown hereafter:

```
ld r1,o(r10)
dadd r3,r1,r2
```

can be replaced by:

```
daddm r3,o(r10),r2
```

New instructions have a CPI = 2 and their use increases the CPI of branch instructions by one. Determine quantitatively the interest of this change considering that the cost of the computer remains the same.

Exercise 1.3.

A program *P* is compiled for a *MIPS R2000* computer working at 100 MHz and integrating a coprocessor. *P* performs two millions floating point operations. The compiler translates each floating point operation into either a coprocessor instruction or a routine of integer instructions attending to the provided set of compilation options. Changing these options, *P* is compiled twice, thus generating two different executables: P_h (code for *MIPS R2000* with coprocessor) and P_s (code for *MIPS R2000* without coprocessor). Both versions are executed and these are the obtained results:

Program	Execution time	Average measured CPI
P_h	92 miliseconds	3.1 cycles
P_s	1.2 seconds	1.2 cycles

Answer the following questions:

1. The number of instructions executed per unit of time for both program versions, expressed in *MIPS* and the total number of executed instructions in each case.

2. The average number of integer instructions replacing each floating point operation in P_s.

3. The throughput of both programs expressed in *MFLOPS*.

Exercise 1.4.

An MP3 compression program has been written in C in order to compare two processors: A (the older one) and B (the new one). Processor A works at 200 MHz. Its instruction set only includes integer instructions and its CPI is 1. On the other hand, processor B works at 900 MHz. Its instruction set extends the A's ISA with multimedia instructions (*MI*). The average CPI provided by processor B varies according to the executed application: integer instructions takes 1 cycle, while multimedia ones take 3 cycles. Depending on the compilation options, the compiler generates either executables containing *MI* instructions or executables coping with the same task but using only integer instructions. This means that each *MI* instruction has an equivalent set of integer instructions for performing the same operation. Two different executables are generated from the compilation of the same compression algorithm: *H* (using *MI* instructions) and *S* (using only integer ones). Results issued from their execution are: processor A needs 50 seconds to compress a song when running code *S* , while processor B performs the same task in 8 seconds when executing code *H*. In addition, it has been observed that with code *H*, the number of executed instructions is 36% less than for code S.

1. Which is the average CPI of processor B when executing *H*?

2. Which is the percentage of *MI* instructions in *H*?

3. How many integer instructions are equivalent to one *MI* instruction?

4. How long will it take for processor B in order to compress the same song running *S*?

Exercise 1.5.

Integer instructions of a CPU execute in 1 clock cycle, while floating point instructions take 5 cycles. Most programs to execute include 20% of floating point instructions. From a performance and cost analysis point of view, to what extent is it interesting to redesign the floating point unit of this CPU to make it 5 times faster if this redesign doubles the cost of the CPU? Justify your answer.

Exercise 1.6.

A computer coprocessor improves the processing of floating point operations by a factor of 5. The execution time of a certain program is 1 minute when the coprocessor is available and it raises to 2.5 minutes without the coprocessor . Compute the percentage of execution time devoted by the program to the execution of floating point operations when the coprocessor is not installed.

Ejercicio 1.7.

A computer provides a 1 Gbps connection. After monitoring the connection, it has been observed that the network manages, in average 50.000 frames per second, each triggering an interrupt request served by the processor running a handler of 200 instructions. The processor works at 200 MHz and its average CPI is of 1.2 cycles.

Answer the following questions:

1. Fraction of time where the processor is serving network interrupts.

2. Improvement resulting from changing the network controller by another reducing the number of interrupt requests to a tenth part. The interrupt handler of this controller requires the execution of 500 instructions. If the computer price is of 2000 €, which should be the maximum price to pay for the replacement in order to satisfy a performance/cost analysis?

Exercise 1.8.

A computer has a load/store processor integrating a single level of (internal) cache L1. Memory access instructions are 30% of the total number of executed instructions. 5% of such memory access instructions lead to cache misses requiring a main memory access that takes 10 clock cycles, which must be added to the time required for serving a cache hit. ThIn absence of cache misses, the processor CPI is 1.

Study the interest of including a second level of (external) cache L2. Experiments performed have shown that L2 solves 90% of L1 misses, thus reducing the penalty induced by such misses to only 2 clock cycles. As a result, only 10% of L1 misses are penalized with 10 clock cycles.

1. Which is the fraction of time spent by the processor accessing main memory when only L1 is available?
2. In average, which is the penalty imposed to the processor in each L1miss when L2 is available?
3. Which is the global speed-up resulting from the inclusion of L2 in the computer?
4. Which is the fraction of time spent by the processor accessing main memory when both L1 and L2 are available?
5. If the computer costs 1000 euros, and adopting a cost-performance point of view, which is the maximum acceptable cost for the processor integrating the second level of cache?

Exercise 1.9.

A program P takes $120\ s$ to execute on a computer C. The following improvements are under evaluation in order to accelerate the execution of this program:

1) Purchase a RAID disk controller card, which would decrease the disk access time to 40% of its original value, reducing at the same time the total execution time of P to $84\ s$. This card costs 90 €.
2) Incorporate a video accelerator card, whose cost is 125 €. This decision is motivated by the fact that P devotes 30% of its execution time to graphics processing. The integration of this accelerator card in the system would reduce the time related to graphics processing to $1/3$ of its original value.

Taking into account these considerations, answer the following questions:

1. If the original computer cost was 300 €, which of the presented improvements are really interesting? Study each option separately from a cost/performance viewpoint. Justify your answer.
2. Which would had been the original cost of computer C to make worthwhile the simultaneous inclusion of both improvements? Adopt a cost/performance viewpoint in your analysis and justify your answer.

Exercise 1.10.

The execution of a given application has been monitored. It has been determined that the application executes in 10 minutes, spending most of its execution time running *P1* and *P2* procedures. For the sake of reducing the application execution time, the code of both procedures is modified. In the ideal case, if the resulting speed-up for both procedures (*P1* and *P2*) is infinite, then the execution time of the application is reduced to 2 minutes.

However, the coded finally produced makes *P1* to carry out its work 5 times faster than before, while *P2* runs 100% faster than originally. In such conditions, the resulting execution time for the application is 4.5 minutes.

Determine the amount of time corresponding to *P1* and *P2* in the original application.

Exercise 1.11.

Farmax Ltd. has a HAL supercomputer devoted to chemical analysis tasks. The computer integrates a RIX/300 processor working at 300 MHz. The system is only used for running the SpectroQuimix program, which makes an intensive use of floating point instructions. After monitoring the system, the following information becomes available:

- The fraction of time devoted by the processor to the execution of floating point instructions is 75%.
- Frequency of floating point instructions in SpectroQuimix is: (see table).
- CPI of each type of floating point instruction is: (see table)

Frequency (%)	Operation	CPI
30	add	5
10	sub	5
40	mult	10
20	div	40

The Farmax direction ask to the head of computer engineers for increasing the computation power of HAL. After checking the alternatives with providers, the following proposal is under study: Replace the processor by the new RIX/400E, which is binary-compatible with the existing processor. It works at 400 MHz and integrates an improved floating point unit. In this new processor, the CPI of integer instructions remains the same, but CPI of floating point ones change as follows:

Operation	CPI
add	3
sub	3
mult	7
div	25

Compute:

1. The speed-up resulting from the execution of integer instructions.
2. The average CPI of floating point instructions of RIX/300 and RIX/400 processors when executing the *SpectroQuimix* program.
3. The throughput (expressed in MFLOPS) resulting from the execution of the *SpectroQuimix* program on the RIX/300 processor.
4. The speed-up resulting from the replacement of the processor when execution floating-point instructions.
5. The global speed-up resulting from the execution of the *SpectroQuimix* program.
6. The fraction of time taken by the RIX/400 processor for the execution of floating-point instructions.
7. The throughput, expressed in MFLOPS, resulting from the execution of the *SpectroQuimix* program on the RIX/400 processor.

Exercise 1.12.

A small Internet server, devoted to service requests arriving through the network, includes a motherboard with two quad-core CPUs based on the AMD Barcelona core. Each CPU has two memory controllers. The application running on the server consists of many independent tasks running concurrently. Each task spends 50% of the execution time executing instructions on the CPU, and 30% of the execution time doing memory accesses that cannot be overlapped with instruction execution (the processor stalls during those accesses).

After several years of operation, the company needs to update the server. Among the available options, there are two configurations: a) a motherboard with two Intel Nehalem processors or b) a motherboard with two AMD Magny-Cours processors. Each Intel processor features 8 cores and 8 memory controllers while each AMD processor features 12 cores and 4 memory controllers. Assuming that all cores in the Intel and the AMD processor are equally fast, that each of those cores is 20% faster than a Barcelona core, and that each memory controller provides the same memory bandwidth and latency for all the processors, which motherboard will deliver the highest speed-up?

Exercise 1.13.

A 500 MHz *l*oad/store processor is modified to add a carry flag in its arithmetic operators. The modification affects both its ALU and instruction set. The analysis of the instruction set and the average *CPI* of the original processor is:

Type	Frequency	*CPI*
arithmetic	50 %	1
load	20 %	2
store	10 %	1.2
branch	20 %	1.2

The modification earns 1 over 10 arithmetic instructions, but their *CPI* increases to 1.2, while the *CPI* of branches goes to 1.5. On the other hand, the redesign of the decoder, the ALU and the hazard detection circuitry have reduced the maximum reachable clock frequency to 400 MHz.

Study the proposed modification according to the following steps:

1. Compute the *CPI* exhibited by the original processor.
2. Compute the *CPI* exhibited by the modified processor.
3. Determine which design is faster and quantify your answer.

Exercise 1.14.

A 32-bit load/store processor is available. The processor integrates instructions to work with *bytes*, *halfwords* and *words*. The execution of $\TeX$ in this processor shows that 11% of data references are performed on *bytes* and *halfwords*, and the rest are carried out on *words*. It is also observed that 36% of program instructions access memory, *load* instructions doubling the frequency of occurrence of *store* ones. Finally, the measured CPI is 1.

Load instructions are the same provided by the *MIPS* instruction set: (*lb*, *lbu*, *lh*, *lhu* and *lw*). The same applies to store instructions (*sb*, *sh* and *sw*).

In order to improve performance, the following architecture modifications are under study:

- Remove from the instruction set those instructions providing access to *bytes* and *halfwords*. As a result, programs requiring the removed instructions will use the following alternatives:

Code with *bytes* access	Code without *bytes* access

`lb/lbu/lh/lhu r,A`	`lw r,A'`
	`extract n,r`

`sb/sh r,A`	`lw r',a'`
	`insert n,r,r'`
	`sw r',a'`

New insertion (*insert*) and extraction (*extract*) instructions can work with either *bytes* and *halfwords* and signed and unsigned data.

- Increase the clock frequency. This improvement can be considered since the previous modification simplifies the design of the interface of the processor core with its cache.

How much should the computer clock frequency be increased to make the incorporation of the aforementioned modifications worthwhile?

Exercise 1.15.

A team of computer architects considers the improvement of the design of the instructions set of a *MIPS*- like computer. The considered modification consist in including in the instructions set instructions to handle the stack: `push` and `pop`. The existence of these new instructions will enable the replacement of sequences of instructions such as:

```
...
sw r1,0[sp]    ; push r1 y r2
sw r2,-4[sp]
subi sp,sp,8
...
lw r4,0[sp]    ; pop r4
addi sp,sp,4
...
```

by the following ones:

```
...
push r1  ; push r1 y r2
push r2
...
pop r4   ; pop r4
...
```

It is known this computer only uses the stack to support subprogram calls. A program call requires passing the incoming parameters and saving and restoring the values of registers used within subprograms for local variable management.

Main program:

```
...
sw r1,0[sp]      ; subprogram call
sw r2,-4[sp]     ; (two parameters)
subi sp,sp,8
jal subprograma
addi sp,sp,8
...
```

Subprogram:

```
; Subprogram entry point
sw r1,0[sp],     ; create space for three
sw r2,-4[sp]     ; local variables
sw r3,-8[sp]
sub sp,sp,#12
...              ; start the code ...
...
lw r3,0[sp]      ; return to the main program
lw r2,4[sp]
lw r1,8[sp]
add sp,sp,#12
jr r31
```

According to results obtained from benchmarking experiments, 1% of instructions are subprogram calls. Regarding the number of local variables and parameters used in subprograms, measures states that:

Loc. var	%	Parámeters	%
0	30	0	45
1	25	1	20
2	20	2	15
3	15	3	10
4	10	4	10

In order to implement the two instructions it is necessary to reduce the clock frequency. It is known that such modification will not impact the average CPI. If in the original design the clock frequency was 100 MHz, which should be the minimum clock frequency of the modified design to make the incorporation of the aforementioned instructions worthwhile?

Exercise 1.16.

Study the interest of including a new indexed addressing mode for *load* and *store* instructions in the *MIPS*. The address of the operand located in memory is computed by adding the contents of two registers and a displacement of 11 bits. As a result the following sequence of instructions:

```
dadd r1,r1,r2
ld rd,o(r1)
```

can be replaced by this other one:

```
ldi rd,o(r1,r2)
```

The CPI is not affected, but the clock period of the improved MIPS is 5% longer than the original. Measures obtained from the original MIPS reflects that 24 % of the executed instructions are of type *load/store*, and 10% of them can be replaced by the new instructions.

Analyze which of both machines is faster and quantify its speed wrt the slowest machine.

Exercise 1.17.

The designers of a computer want to evaluate the pertinence of two alternatives for coding conditional branches in programs:

- 1 A Comparison instruction (1 cycle) followed by a branch instruction (2 cy- cles). 2.

- 2 Comparison+Branch instruction(2 cycles).This alternative increases the clock period in 25% of its original value.

Assuming that branches are 20% of all executed instructions in the first alternative, and the rest of instructions execute in 1 cycle. Which is the best alternative?

Ejercicio 1.18.

MIPS64 SIMD extensions are known as *MIPS64 SIMD Architecture* (MSA). They add a new register file including 32 registers of 128-bit each, named `w0`, `w1`, etc. Depending on the instruction, each register can be handled as a 16-byte vector (8 bits per vector element), as a vector of 8 *halfwords* (16 bits per vector element), as a 4-*words* vector (32 bits per element), or as a 2-*doublewords* vector (64 bits per element).

In order to specify the type of element, SIMD instructions should include one of the following sufixes: `.b`, `.h`, `.w`, or `.d`; for referring to bytes, halfwords, words, and doublewords, respectively. So, the instruction `addv`**`.w`** `w5,w1,w2` adds all 4 **word** elements in vector registers `w1` and `w2` (the first element of `w1` is added to the first element of `w2`, the second one to the second one, and so on). Then, the resulting vector components are stored into register `w5`.

MSA extensions wants to be used in order to accelerate the execution of the following algorithm, which computes the vector operation $\vec{z} = A \cdot \vec{x} + \vec{y}$:

```
    addi r10,r0,N    ; N is the vector size
    addi r11,r0,0
    addi r12,r0,A
loop:
    lw r20,X(r11)
    lw r21,Y(r11)
    mul r20,r20,r12
    add r21,r21,r20
    sw r21,Z(r11)
    addi r11,r11,+4
    addi r10,r10,-1  ; 1 iteration processes 1 element of vectors x, y, z
    bne r10,r0,loop
```

In order to cope with that goal two versions of the original algorithm, both using MSA extensions, are implemented. The first one (MSA1) executes the following code (MSA instructions are bolded):

```
    addi r10,r0,N    ; N is the vector size
    addi r11,r0,0
    ldi.w w12,A
loop:
    ld.w w20,X(r11)
    ld.w w21,Y(r11)
    mulv.w w20,w20,w12
    addv.w w21,w21,w20
    st.w w21,Z(r11)
    addi r11,r11,+16
    addi r10,r10,-4  ; 1 iteration processes 4 element of vectors x, y, z
    bne r10,r0,loop
```

Considering that each loop iteration in MSA1 processes 4 elements (in stead of only 1 in the original algorithm), answer the following questions:

1. Assume N is a multiple of 4, how many instructions does MSA1 execute in total?

2. Considering that the CPI of all instructions is 1 and MSA extensions do not affect the processor frequency, which is the speed up resulting from using MSA1 in stead of the original algorithm?

3. A second version of the algorithm (MSA2) improves MSA1 by using a the instruction maddv, which La segunda versión (MSA2) mejora MSA1 mediante el uso de la instrucción `maddv`, which combines instructions `mulv.w` and `addv.w`. As a result, the following code:

   ```
   mulv.w w20,w20,w12
   addv.w w21,w21,w20
   ```

   becomes:

   ```
   maddv.w w21,w20,w12
   ```

   However, the CPI of the instruction `maddv.w` is 3 due to data hazards with `ld.w` instructions, and the fact that the instruction carries out with 2 operations. Attending to this, how much should be increased the processor frequency to make the execution of MSA2 better than the one of MSA1?