Name, SURNAME and ID ⇒

Middle East Technical University
Department of Computer Engineering

CENG 331

Section 2,3
Fall '2018-2019
Final

- **Duration: 120** minutes.
- **Exam:**
  - This is a **closed book**, **closed notes** exam.
  - No attempts of cheating will be tolerated. In case such attempts are observed, the students who took part in the act will be prosecuted. The legal code states that students who are found guilty of cheating shall be expelled from the university for **a minimum of one semester**!
  - Appendix contains the reference material that you can use during the exam. **Do not mark the Appendix. It will not be graded!**
  - **Use the space given on the exam papers. No extra papers will be handed and back of the exam papers will not be graded!**
  - **This booklet consists of *9* pages including this page. Check that you have them all!**
  - **Write your name on ALL PAGES!**

Question 1	
Question 2	
Question 3	
Question 4	
Question 5	
Total ⇒	

Name, SURNAME and ID ⇒

1 (10 pts)

Answer the following questions:

(a) (1 pt) Write the binary representation for `TMax+TMax` in a system where `int`'s are represented as 4 bytes.

(b) (2 pt) What's the main difference between a bubble and a nop instruction?

(c) (2 pts) Consider a virtual memory system that uses single-level page table to translate virtual addresses to physical addresses. How would the number of bits in each PTE (page table entry) would change (a) if the physical memory size is doubled, (b) if the virtual memory size is doubled?

(d) (1 pts) How is an ISA (Instruction Set Architecture) defined?

(e) (2 pts) What does a Translate Lookahead Buffer (TLB) translate? Be precise.

(f) (2 pts) Float representation uses 1 bit for sign, 8-bits for exp and 23 bits for frac. If a float has `1|11111110|00000000000000000000001` as its binary representation, what is the value it represent? Write your result in the form of $S * M * 2^E$ where S, E, M are decimal numbers.

exp = 62 bias = 63 exp-bias = -1 = E

M = 1.frac = $1 + 1 \times 2^{-23}$

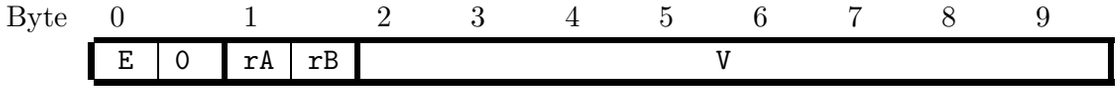
$(-1)^0 * \left(\frac{2^{23}+1}{2^{23}}\right) * 2^{-1}$

Name, SURNAME and ID ⇒

2 **(30 pts)** Consider the SEQ processor designed in our lectures (see the Appendix for the HCL code and schematics). Suppose we want to add a new instruction

```
leaq (rA, V), rB
```

which computes `V+Reg[rA]` and stores the result in `rB`. The instruction is encoded as



Fill out the following form to describe what needs to happen during each stage of this new instruction.

(a) (10pts) Write down the stages of execution.

Stage	Computation
Fetch	icode : ifun ← M_1[PC] rA : rB ← M_1[PC+1] valC ← M_8[PC+2] valP ← PC+10
Decode	valA ← R[rA]
Execute	valE ← valA + valC
Memory	—
Write-back	R[rB] ← valE
PC-update	PC ← valP

Name, SURNAME and ID ⇒

(b) (20pts) Write down the HCL code that needs to be changed using the SEQ HCL code in the appendix as a reference.

```
wordsig ILEQ 'I_ILEQ'
bool instr_valid = icode in { .......,  ILEAQ };
```

✓ bool need_regids

✓ bool need_valC

✓ word srcA =
    icode in {ILEAQ} : rA;

✓ word dstE =
    icode in {ILEAQ} : rB;

✓ word AluA :
    valA

✓ AluB valC

✓ new PC
    val+

Name, SURNAME and ID ⇒

3 **(10 pts)** Consider the PIPE+ processor, whose overall architecture and HCL code is provided in the appendix. The following code sequence is being executed:

```
0x000 irmovq 0x100, %rax
0x006 mrmovq (%rbx), %rsp
0x00c popq %rax
0x00e xorq %rdx, %rdx
0x010 jne 0x000
0x015 movq %rdx, %rax
```

Note that all conditional branches are assumed to be taken and forwarding is implemented.

Identify the data and control hazards in the code above. Specifically discuss the type of each hazard and write down the registers/instructions involved, comment on how PIPE+ architecture deals with them.

Name, SURNAME and ID ⇒

4 **(20 pts)** Consider the following C function to sum all the elements of a 5×5 matrix. Note that it is iterating over the matrix column-wise.

```
char sum_matrix(char matrix[5][5]) {
    int row, col;
    char sum = 0;
    for (col = 0; col < 5; col++) {
        for (row = 0; row < 5; row++) {
            sum += matrix[row][col];
        }
    }
    return sum;
}
```

Suppose we run this code on a machine whose memory system has the following characteristics:

- Memory is byte-addressable.
- There are registers, an L1 cache, and main memory.
- A char is stored as a single byte.
- The cache is direct-mapped, with 4 sets and 2-byte blocks.

You should also assume:

- `matrix` begins at address `0`.
- `sum`, `row` and `col` are in registers; that is, the only memory accesses during the execution of this function are to `matrix`.
- The cache is initially cold and the array has been initialized elsewhere.

Fill in the table below. In each cell, write "h" if there is a cache hit when accessing the corresponding element of the matrix, or "m" if there is a cache miss.

	0	1	2	3	4
0					
1					
2					
3					
4					

Name, SURNAME and ID ⇒

5 **(30 pts)** This problem deals with virtual memory address translation using a multi-level page table, in particular the 2-level page table for a 32-bit Intel system with 4 KByte page tables. The following diagrams are direct from the Intel System Programmers guide and should be used on this problem. Note that the first-level page table is referred to as "Page Directory" and its entries are as "Directory Entry". The PTBR (Page Table Base Register) appears as CR3 (Page Directory Base Register).

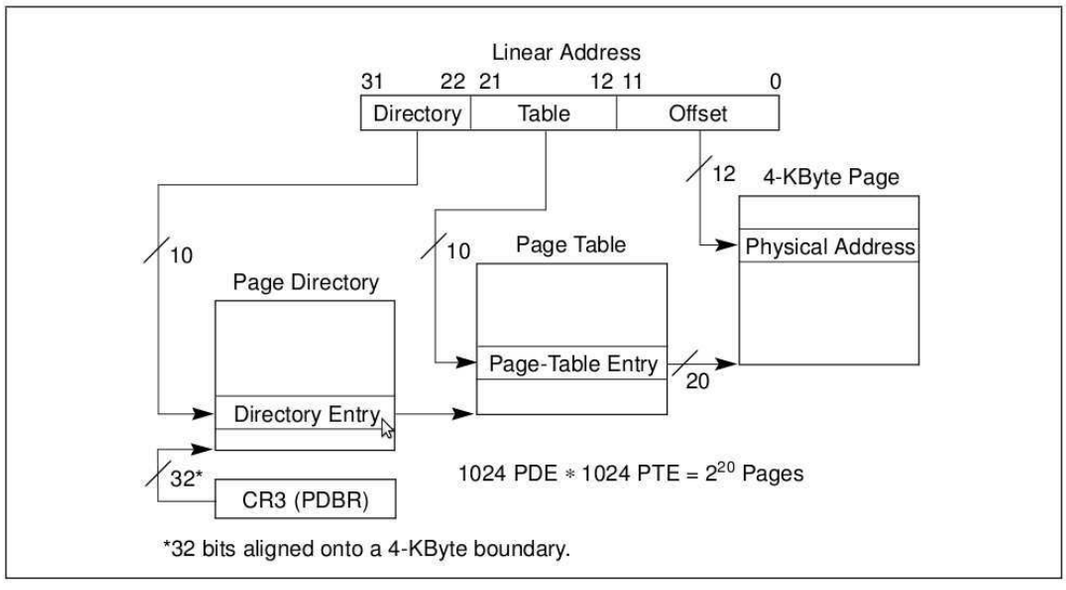


Figure 3-12. Linear Address Translation (4-KByte Pages)

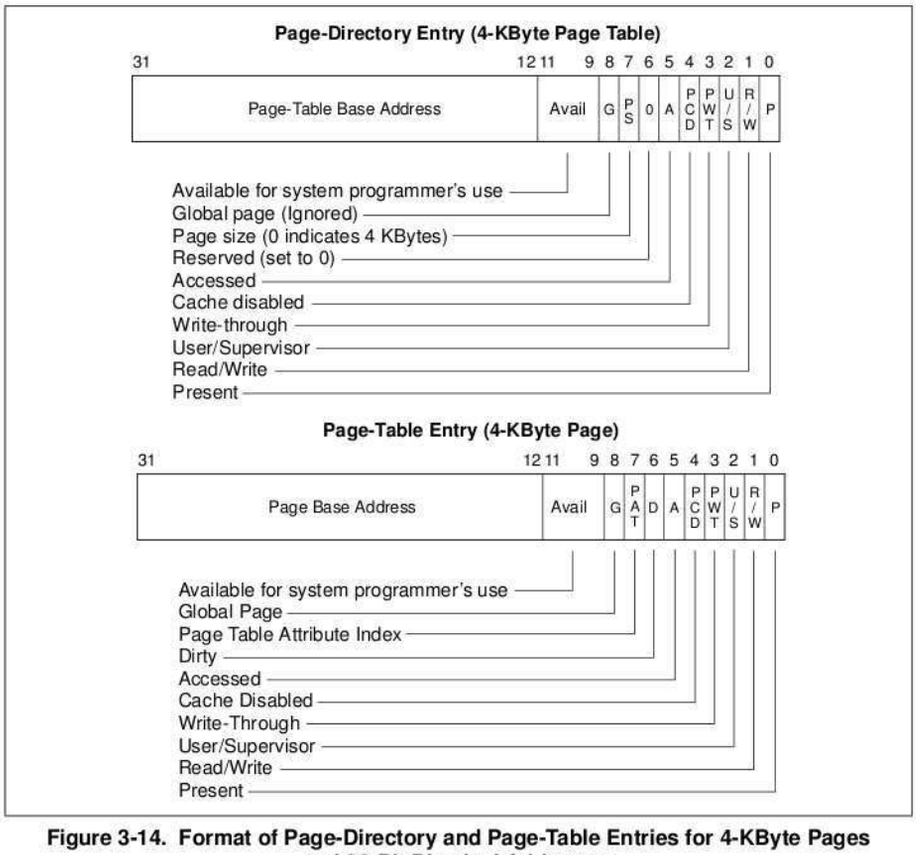


Figure 3-14. Format of Page-Directory and Page-Table Entries for 4-KByte Pages and 32-Bit Physical Addresses

Name, SURNAME and ID ⇒

The contents of the relevant sections of memory are shown on this page. All numbers are given in hexadecimal. Any memory not shown can be assumed to be zero. The Page Directory Base Address (PDBR) is `0x0c23b000`.

For each of the following problems, perform the virtual to physical address translation. If an error occurs at any point in the address translation process that would prevent the system from performing the lookup, then indicate this by circling FAILURE and noting the physical address of the table entry that caused the failure.

For example, if you were to detect that the present bit in the PDE is set to zero, then you would leave the PTE address in (b) empty, and circle FAILURE in (c), noting the physical address of the offending PDE.

Address	Contents
00023000	beefbee0
00023120	12fdc883
00023200	debcfd23
00023320	d2e52933
00023fff	bcdeff29
00055004	8974d003
0005545c	457bc293
00055460	457bd293
00055464	457be293
0c23b020	01288b53
0c23b040	012aab53
0c23b080	00055d01
0c23b09d	0ff2d303
0c23b274	00023d03
0c23b9fc	2314d222
2314d200	0fdc1223
2314d220	d21345a9
2314d4a0	d388bcbd
2314d890	00b32d00
24aee520	b58cdad1
29de2504	56ffad02
29de4400	2ab45cd0
29de9402	d4732000
29dee500	1a23cdb0

THIS PAGE WILL NOT BE GRADED!
Write your answers on the next page!

Name, SURNAME and ID ⇒

(a) (15 pts) Read from virtual address `0x080016ba`.

- (i) Physical address of PDE:
- (ii) Physical address of PTE:
- (iii) (SUCCESS) The physical address accessed is:

  OR

  (FAILURE) The physical address of the table entry causing the failure is:

(b) (15 pts) Read from virtual address `0x9fd28c10`.

- (i) Physical address of PDE:
- (ii) Physical address of PTE:
- (iii) (SUCCESS) The physical address accessed is:

  OR

  (FAILURE) The physical address of the table entry causing the failure is: