Assignment 1

due Tuesday, Jan 22 at the 10:30 lecture

January 5, 2019

Student Full Name:

Student Number:

Group Number:
1 COMPUTER ARCHITECTURE QUESTIONS

1. Consider a computer system that uses a 32-bit address bus and a 16-bit data bus. Then:
   (i) How many memory locations can the computer address? (ii) How many bits are per
   address location? and (iii) What is the maximum size of the main memory?
   i. ____________________________________________
   ii. ____________________________________________
   iii. ____________________________________________

2. The MDR (Memory Data Register) and the MAR (Memory Address Register) are two
   important internal register of a microprocessor. Explain when the microprocessor uses
   these two registers?
   _______________________________________________________________________
   _______________________________________________________________________

3. Consider the following x86-based assembly code:
   
   [0x153F]   Mov Ax, 5
   [0x1540]   Mov Bx, 3
   [0x1541]   Mul Ax, Bx
   [0x1542]   Xor Ax, Ax
   [0x1543]   Halt

   a) If the CIR (Current Instruction Register) contains the instruction stored in memory
      address 0x1540, then what is the value of the IP (Instruction pointer) and Ax register?
      _______________________________________________________________________

   b) If the decoder unit has just finished decoding the instruction stored in memory ad-
      dress [0x1542] then what is the value of the IP (Instruction pointer), Ax register, and
      Bx register?
      _______________________________________________________________________

Copyright © Karim Lounis, 2019.
4. Write a C-program (only some instruction statements) for the following two x86-based assembly codes (note: the program can be written in different ways):

```
[0x153F] Mov Ax, 1
[0xB53F] Mov Ax, 1
[0x1540] X Nop
[0xB540] Jmp X
[0x1541] Cmp Ax, 1
[0xB541] Y Nop
[0x1542] Je X
[0xB542] Je Y
[0x1543] Halt
[0xB543] Halt
```


6. Can a uniprocessor computer be a multiprogramming system? Explain.

7. Give an example of a multitasking and single-tasking operating system?

Copyright © Karim Lounis, 2019.
8. What is the difference between a distributed computing system and a clustered computing system?
____________________________________________________________________
____________________________________________________________________

9. What is the program that is responsible for booting the operating system? where is that program stored?
____________________________________________________________________
____________________________________________________________________

10. What is a system call? give one example.
____________________________________________________________________
____________________________________________________________________

2 PROCESSES, INTERRUPTS, AND CONTEXT SWITCHING

1. Give three reasons that may lead a program to be switched into the suspended state.
   i. _____________________________________________________________________
   ii. ___________________________________________________________________
   iii. ___________________________________________________________________

2. The operating system uses a data structure to represent a process and manipulate it, what is that structure? Give five information fields that are kept in that data structure.
____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
____________________________________________________________________

Copyright © Karim Lounis, 2019.
3. How does the operating system locates that data structure of a given process?

4. What is the Zombie state of a process in Unix based systems? How can you simulate that in a C-program?

5. What is an orphan process in UNIX-like operating systems?

6. What is a context switching? and why is that needed?

7. What is an interrupt? and what is its relation with context switching?

8. Can we have an interrupt without context switching?
9. Explain what does the `fork()` function call perform in a C-program running on a UNIX-like system? What happen when a process executes the `fork()` function?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

10. What will happen when the ALU (Arithmetic and Logical Unit) starts executing the instruction at the address 0x330F in the following program?

```
[0x330B]  Mov  Ax, 0x000A
[0x330C]  Add  Ax, 0x0002
[0x330D]  Mul  Ax, 0x000A
[0x330E]  Xor  Ax, Ax
[0x330F]  Div  Ax
[0x3310]  Halt
```

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

11. Consider the following x86-based assembly code that is currently executing on a computer that has a memory which is Byte-addressable and a CPU that has a word size of 2-Bytes:

```
Mov  Ax, [0X3F55]  (encoded in 2 words)
Mov  Bx, [0X3F56]  (encoded in 2 words)
Mul  Cx, 0x0000    (encoded in 1 word)
Xor  Bx, Cx        (encoded in 1 word)
Inc  Ax             (encoded in 1 words)
Halt               (encoded in 1 words)
```

Copyright © Karim Lounis, 2019.
i) Assuming that the address of the first instruction in this program is 0xFBC0 and that an interrupt occurs after the completion of the 4th-instruction i.e., `Xor Bx, Cx`, what will be the value of the IP register that will be pushed onto the stack?

ii) Which components is responsible for pushing the PC value into the stack? why?

iii) Which stack are we referring to?

12. Explain why pooling is not a good mechanism for receiving hardware events.

13. What is the difference between software-based interrupts and hardware-based interrupts? Give one example for each type of interrupt.

14. Give the pseudo-code using `(Begin-End)` for sequential executions and `(ParBegin-ParEnd)` for parallel executions, for the following PPG (Process Precedence Graph) where $S_0$ is the first statement:
15. If the program (which you just constructed above) runs on single-core CPU system then: Do the blocks that are supposed to be executed in parallel executes in real concurrency?

16. Does the PC increments (updates its value) after or before the current instruction is decoded? Provide two arguments to support your answer.

17. In the following program, a parent process executes the primitive `fork()` to create a child process. Initially, the variable `x` has the value 13, the child process code snippet has been implemented in such a way it increments the value of `x` by 7, and the parent process displays the value of `x` after its child process terminates. Explain the value displayed by the parent process.

Copyright © Karim Lounis, 2019.
In the following program, give the final number of processes that display the `printf` message (Caution: do not execute that program as is. It may crash your computer).
```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void main()
{
    for(int i=1001; i>0; i--)
    {
        fork();
    }
    printf("Hello this is process[%d] writing ...
", getpid());
    return;
}
```