Birkbeck
(University of London)

BSc/MSc Examination

Department of Computer Science and Information Systems

MOCK EXAM (COSS/CompSys)

There are five questions in this paper; each of them is compulsory and worth 20 marks.
The paper is not prior-disclosed.
The use of additional material (e.g., electronic calculators) is not permitted.

1. Consider the computation `(M1+M2)*(M3+M4)` where `M1`, ... denote the content of memory locations.

   (a) Write assembly code typical of RISC machines for this computation. Use at most two operands for each instruction (e.g., `ADD r1 r2` for `r1 <- r1 + r2`) and use at most three registers altogether. The result of the computation should be stored in a register.

   (6 marks)

   (b) Apply the register renaming technique to remove the false dependencies from the code in item (a). Explain which false dependencies have been removed.

   (4 marks)

   (c) Show the pipeline activity by drawing a diagram when the assembly code from item (b) is executed on a superscalar processor. There are four pipeline stages: fetch–decode, register-read, execute and write-back. There are two functional units for each pipeline stage and there is an instruction window with capacity of two instructions. Explain when instructions can skip pipeline stages and where delay slots occur. State you additional assumptions.

   (10 marks)

   Marking scheme for Question 1: Marks will be deducted for incorrect code (e.g., not loading the data into registers), for introducing unnecessary registers, for violating data dependencies, for using unnecessary pipeline stages and for missing opportunities for out-of-order execution.

2. (a) For every **register** below

PC	program counter
IR	instruction register
MAR	memory address register
MBR	memory buffer register
PSW	program status word

find its correct **description** from this list:

i. Contains status information about the processor and the currently running process.
ii. Contains the next instruction to be executed.
iii. Used for transferring addresses between the CPU and main memory.
iv. Contains the instruction being currently executed.
v. Contains the address of the instruction being currently executed.
vi. Used for storing temporary data.
vii. Contains the address of the next instruction to be executed.
viii. Used for computing complex (indirect) addresses.
ix. Used for transferring data between the CPU and main memory.
x. Contains the context of the previously running process.

(5 marks)

Marking scheme for Question 2(a): +1 mark for each correct definition and -1 mark for each incorrect definition, with the adjustment that the overall mark is at least 0.

(b) Replace the numbers [**n**] in the text by the appropriate words/expressions from the list below. Note that some words/expressions may correspond to several numbers or no number at all.

Words/Expressions: control, dependencies, different, general purpose, identical, life span, main memory, physical, temporal, virtual.

The following text describes the compiler-based register optimization technique: The compiler allocates separate [**1**] registers to the variables and other quantities occurring in the code. Then the compiler analyses the code to find out the [**2**] of each [**3**] register. Then it tries to allocate a [**4**] register to each [**5**] register using the following strategy. When the [**6**] of two [**7**] registers overlap the [**8**] register(s) allocated to them must be [**9**]. On the other hand, when the [**10**] of two [**11**] registers are disjoint the [**12**] register(s) allocated to them can be [**13**]. If the number of [**14**] registers is not sufficient, then some [**15**] registers must be saved in [**16**].

(8 marks)

Marking scheme for Question 2(b): +0.5 mark for each correct replacement and -0.5 mark for each incorrect replacement (0 mark for no replacement), with the adjustment that the overall mark is at least 0.

(c) Replace the numbers [**n**] in the text by the appropriate words/expressions from the list below. Note that some words/expressions may correspond to several numbers or no number at all.

Words/Expressions: control, general purpose, life span, local, main memory, parameter(s), physical, procedure(s), temporary, virtual.

The following text describes register windows: Register windows are subsets of [**1**] registers allocated to [**2**]. Every register window is divided into three parts. The [**3**] registers contain data that has been passed on by another [**4**]. The [**5**] registers contain [**6**] data. Finally, [**7**] registers contain [**8**] that the [**9**] passes on to another [**10**]. These parts are arranged so that the [**11**] registers of the main [**12**] coincide with the [**13**] registers of a called [**14**].

(7 marks)

Marking scheme for Question 2(c): +0.5 mark for each correct replacement and -0.5 mark for each incorrect replacement (0 mark for no replacement), with the adjustment that the overall mark is at least 0.

3. Consider the following attempt to solve the critical section problem for two processes where the initial value of turn is 0 or 1:

```
while(TRUE){                    while(TRUE){
   while(turn!=0);                 while(turn!=1);
   critical-section();             critical-section();
   turn=1;                         turn=0;
   noncritical-section();          noncritical-section();
}                               }
```

Process 0 Process 1

(a) For each **concept** below

- race condition,
- critical section,
- mutual exclusion,

find the correct **definition** from the following list:

i. The situation when several processes compete for the same resource.
ii. The situation when the first process scheduled can get hold of the resources.
iii. The situation when the outcome of a computation depends on the relative speed of the processes.
iv. A section of code where a resource is accessed.
v. A section of code where a process communicates with another process.
vi. A section of code where only one process can be in.
vii. The requirement that a process outside of its critical section cannot prevent another process to enter its critical section.
viii. The requirement that only one process can be in the critical section.
ix. The requirement that only one process can hold any of the resources.

(6 marks)

Marking scheme for Question 3(a): +2 marks for each correct definition and -2 marks for each incorrect definition, with the adjustment that the overall mark is at least 0.

(b) A simple **True** or **False** answer will suffice for the following items in this question.

i. The above code is deadlock free.

(2 marks)

ii. The above code achieves mutual exclusion.

(2 marks)

iii. The two processes can go into critical section only in an alternating way.

(2 marks)

iv. The above code satisfies the condition *progress*.

(2 marks)

v. The above code is starvation free.

(2 marks)

vi. Replacing 'while(turn!=0);' with 'while(turn=1);' and 'while(turn!=1);' with 'while(turn=0);' simultaneously in the code would result in deadlock.

(2 marks)

vii. Replacing 'while(turn!=0);' with 'while(turn=0);' and 'while(turn!=1);' with 'while(turn=1);' simultaneously in the code would result in starvation.

(2 marks)

Marking scheme for Question 3(b): +2 marks for each correct answer and -2 marks for each incorrect answer, (0 mark for no answer) with the adjustment that the overall mark is at least 0.

4. (a) Replace the numbers [**n**] in the text by appropriate words/expressions.

The following text describes the overhead related to process switches: When a process switch occurs the system has to save the [**1**] of the process that has been running. This includes saving the contents of the [**2**], [**3**] and the other [**4**]. Then the [**5**] block of the process in the [**6**] table has to be updated, e.g., changing the state of the process from [**7**] to either [**8**] or [**9**]. Then the [**10**] code is executed to choose the next [**7**] process. The [**5**] block of the chosen process has to be updated and its [**1**] has to be restored on the CPU so that its execution can resume.

(10 marks)

Marking scheme for Question 4(a): +1 mark for each correct replacement and -1 mark for each incorrect replacement (0 mark for no replacement), with the adjustment that the overall mark is at least 0.

(b) Consider a (RISC) machine that uses 4 KB pages and 4-byte instructions, and assume that a process switch just occurred. If the instruction is in main memory, it takes 5 ns to load it into the IR. Loading a page from the hard disk into memory takes 10 ms. The initial probability (after the process switch) that a referenced instruction is in main memory is 0.2. How many nanoseconds does it take to load the first referenced instruction into the IR on average?

(6 marks)

(c) Explain how the situation changes for loading the second instruction, and give an estimate for the average load time in microseconds for this case. (You can assume that the first instruction referenced is not a jump.)

(4 marks)

Marking scheme for Question 4(b),(c): Marks will be deducted for flaws in the computations and for arithmetic errors.

5. (a) Replace the numbers **[n]** in the text by appropriate words/expressions. Different numbers may refer to the same word/expression.

The following text describes the way a process can access system resources: When a process wants to access a system resource it issues a **[1]**. The OS first checks whether the process has access **[2]** to the resource by checking either the process' **[3]** list or the resource's **[4]** list. Provided that the process has the correct access **[5]** to the resource, the OS checks whether the resource is **[6]** or it has been already **[7]** to another process. In the first case, the OS **[8]** the resource to the process, and in the second case, the OS **[9]** the process.

(9 marks)

Marking scheme for Question 5(a): +1 mark for each correct replacement and -1 mark for each incorrect replacement (0 mark for no replacement), with the adjustment that the overall mark is at least 0.

(b) A system has four processes p_1, p_2, p_3, p_4 and three types of dedicated resources R_1, R_2, R_3. The existence vector is $E = (6, 4, 4)$.

- Process p_1 holds two units of R_1 and requests two units of R_2;
- Process p_2 holds three units of R_2 and requests three units of R_1 and two units of R_3;
- Process p_3 holds two units of R_1 and requests two units of R_2;
- Process p_4 holds four units of R_3 and requests two units of R_1.

i. Compute the availability vector.

(2 marks)

ii. Explain whether the system is deadlocked.

(4 marks)

iii. Determine whether this state of the system is safe.

(5 marks)

Marking scheme for Question 5(b): Marks will be deducted for incorrect answers and for flaws in the arguments.