Chalmers University of Technology and Gothenburg University

Operating Systems
EDA092, DIT 400

Exam 2016-01-13

Date, Time, Place: Wednesday 2016/01/13, 08:30-12:30, "Maskin"-salar

Course Responsible: Vincenzo Gulisano, Marina Papatriantafilou

Auxiliary material: You may have with you
- An English-Swedish, Swedish-English dictionary.
- No other books, notes, calculators, PDA's etc.

Grade-scale ("Betygsgränser"):
CTH: 3:a 30-39 p, 4:a 40-49 p, 5:a 50-60 p
GU: Godkänd 30-49p, Väl godkänd 50-60 p

Exam review ("Granskningstid"):
Will be announced after the exam.

Instructions

- Do not forget to write your personal number, if you are a GU or CTH student and at which program ("linje").
- Start answering each assignment on a new page; number the pages and use only one side of each sheet of paper.
- Write in a **clear manner** and **motivate** (explain, justify) your answers. If it is not clear what is written, your answer will be considered wrong. If it is not explained/justified, even a correct answer will get **significantly** lower (possibly zero) marking.
- If you make **any assumptions** in answering any item, do not forget to clearly state what you assume.
- The exam is organized in groups of questions. The credit for each group of questions is mentioned in the beginning of the respective group. Unless otherwise stated, all questions in a group have equal weight.
- Answer questions in English, if possible. If you have large difficulty with that and you think that your grade can be affected, feel free to write in Swedish.

Good luck !!!!

1. (12 p)

   (a) (4p) Describe what base and limit registers are, what they are used for and why setting them requires privileged instructions.

   HINT: Please refer to slides 10-12 of Lecture 7 and the respective sections in the course book.

   (b) (4p) Given the following segment table:

Segment	Base	Length
0	117	500
1	2510	250
2	3600	50
3	1270	650

   tell which physical addresses would be accessed for the following logical addresses:
   0, 400
   3, 600
   2, 60
   1, 100
   2, 49
   0, 550
   3, 700
   0, 0
   HINT:
   517
   1870
   none (illegal reference)
   2610
   3649
   none (illegal reference)
   none (illegal reference)
   117

   (c) (4p) Discuss why paging suffers from internal fragmentation and how the page size can affect internal fragmentation.

   HINT: Please refer to slide 49 of Lecture 7 and the respective section in the course book.

2. (12 p)

   (a) (4p) Given the following reference string, compute the minimum number of page faults that is observed if 3, 4 or 5 frames (all free in the beginning) are available.

   721201132470072137030003

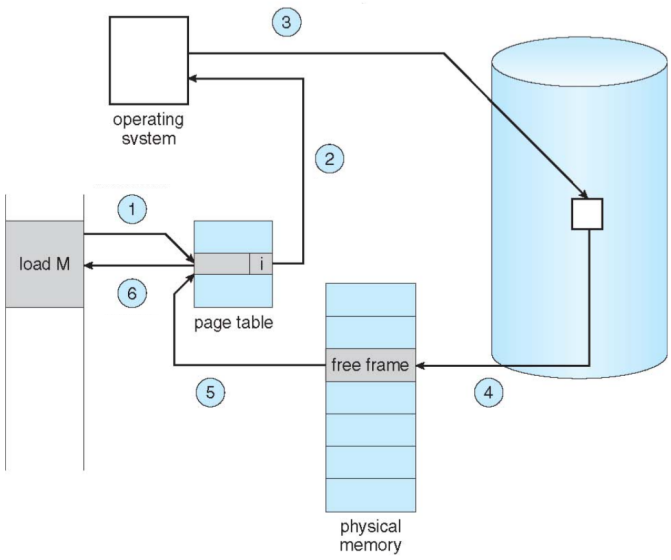
   HINT: 9,8 and 7

   (b) (4p) Describe what the following figure (taken from Operating System Concepts, Silberschatz, 2013) represents and discuss each step separately.

   HINT: Please refer to slides 16-17 of Lecture 8 and the respective sections in the course book.

   (c) (4p) What is trashing and what is the relation between processes' localities and trashing?

   HINT: Please refer to slides 56-60 of Lecture 8 and the respective sections in the course book.



3. (12 p)

   (a) (4p) Given Amdahl's law, calculate the minimum number of cores needed for an application that has a 50% parallel component to observe a speed up greater than or equal to 1.9.
   HINT: 19

   (b) (4p) Explain the difference between concurrent and parallel execution of a process.
   HINT: Please refer to slides 13 and 18 of Lecture 3 and the respective sections in the course book

   (c) (4p) Describe the two main models of interprocess communication.
   HINT:Please refer to slides 41,43 and 45 of Lecture 2 and the respective sections in the course book

4. (12 p)

   (a) (4p) Why two interrupt request lines exist and how are they used?
   HINT: Please refer to slides 23-25 of Lecture 10 and the respective sections in the course book

   (b) (4p) Describe the mechanism of Direct Memory Access, why is it beneficial?
   HINT: Please refer to slides 27-29 of Lecture 10 and the respective sections in the course book

   (c) (4p) Given the following list of entries (each composed of the read / write / execute permissions for owner, group and other), the owner (hidden), the group and the name of the file:

   - *r w x r w x r - x ? groupC file1*
   - *r w x - - x - w x ? groupA file2*
   - *r w x - - x - - - ? groupB file3*
   - *r w - r - x - w - ? groupC file4*

and given the following list of operations (marked as successful or failed) performed by userX

- delete file1 (successful)
- read from file2 (successful)
- read from file3 (failed)
- write to file3 (failed)
- write to file4 (successful)

Specify, for each file, whether userX owns the file, does not own the file or whether you cannot say. Also, for each group, specify whether userX belongs to the group, does not belong to the group or whether you cannot say.

HINT: file1 owns, file 2 owns, file 3 owns not, file 4 cannot say. groupA cannot say, groupB cannot say, group C does not belong.

5. (12 p)

   (a) (8p) Consider a system with n threads that need mutually exclusive access to a single resource in the system. The system provides the possibility to execute atomically TestAndSet on shared variables, i.e. upon invocation of a TestAndSet, the following is executed atomically by the hardware:

```
boolean TestAndSet (boolean *target){
  boolean rv = *target;
  *target = TRUE;
  return rv} // Executed atomically by the HW
```

Can the following solve the resource allocation problem when the system is (i) uniprocessor or multiprocessor/multicore with preemptive scheduling (ii) uniprocessor with non-preemptive priority scheduling?
In each case explain carefully why or why not.

```
shared boolean V, initialized to FALSE;

// pseudocode for each thread i:

while(true){
  while(TestAndSet(&V)); //busywait
     // access the resource
  lock = FALSE;
     // remainder section //
}
```

HINT: (i) in a preemptive scheduling system, among all competing threads, one will eventually succeed in its TestAndSet when V is (or becomes) false, thus guaranteeing mutual exclusion and progress as needed; fairness is not guaranteed though, some threads might suffer starvation if is unlucky and only gets dispatched when V is TRUE; (ii) in a uniprocessor system with non-preemptive priority scheduling, it is possible to have no-progress, if V is true due to a low-priority thread i accessing the resource, and a high-priority thread j is spinning, not allowing i to finish accessing the resource and change V to false

(b) (4p) Consider a uniprocessor system and a set of n periodic real-time tasks, each of them with given period p_i and processing requirement t_i in each period interval. Do we have a method to decide whether the set is schedulable in the system? If yes, explain the method and give an example of a schedulable task set and an example of non-schedulable task set; if no, explain why not.

HINT: task set is schedulable iff EDF scheduling meets all deadlines, ie iff utilization (sum of t_i/p_i for all i is smaller than 1)