8-hour Take-Home Exam in Computer Systems

Department of Computer Science, University of Copenhagen (DIKU)
Date: January 27, 2021

Preamble

This is the exam set for the 8 hour written take-home exam in Computer Systems (CompSys), B1+2-2020/21. This document consists of 11 pages excluding this preamble; make sure you have them all. Read the rest of this preamble carefully. Your submission will be graded as a whole, on the 7-point grading scale, with external censorship.

I addition to this document you are given two templates to write your answer. One is formatted as a LaTeX document, while the other is a pdf-file with included form fields. You can use either of these to make your answer.

- You can answer in either Danish or English.
- Remember to write your exam number (and only your exam number) on the hand-in.
- Do not exceed the line limits indicated in the answer-templates (see below).
- Only hand-in one pdf-file with your answers on eksamen.ku.dk. Note: you can only hand-in once.

Expected usage of time and space

The set is divided into sub-parts that each are given a rough guiding estimate of the size in relation to the entire set. However, your exact usage of time can differ depending on prior knowledge and skill.

Furthermore, all questions in the answer-templates includes limitation to the number of lines for answers in a standard page and font formatting. These limitations are more than enough to include a complete and satisfactory answer of the question; thus, full answers of the question does not necessarily use all available space.
These limitations are strict, meaning we reserve the right to *not* evaluate any part of the answer exceeding the given limits.

Exam Policy

This is an *individual*, open-book exam. Thus, you are not allowed to discuss the exam set with anyone else, until the exam is over (be aware of co-students that can have gotten extra time). It is expressly forbidden:

- To discuss or share any part of this exam, including, but not limited to, partial solutions, with anyone else.
- To help or receive help from others.
- To seek inspiration from other sources, including the Internet, without proper citation in your answer.
- Copy-paste a text (even if it partly edited) without proper citation in your answer.
- To post any questions or answers related to the exam on *any fora*, before the exam is over, including the course discussion forum on Absalon and Discord.

Exam number: _____

Breaches of this policy will be handled in accordance with the standard disciplinary procedures. Possible consequences range from your work being considered `void`, to *expulsion from the university*[1].

You may use the course book, notes and any documents printed or stored on your computer and other device. If you use any other material, you must cite this in your answer.

Errors and Ambiguities

In the event of errors or ambiguities in the exam text, you are generally expected to state your assumptions as to the intended meaning in your answer. Some ambiguities may be intentional.

If you are in doubt, you can contact `m.kirkedal@di.ku.dk` for clarification. But you will not get answers to clarification of course curriculum.

If there during the exam are corrections or clarifications to any question, these will be posted as an announcement on Absalon course page. Be sure that you receive notifications from this.

[1] `https://uddannelseskvalitet.ku.dk/docs/overordnede-dokumenter/Ordensregler-010914.pdf`

8-hour Take-Home Exam in Computer Systems
Department of Computer Science, University of Copenhagen
Date: January 27, 2021

Exam number: _____

1 Machine architecture (about 33 %)

1.1 Assembler programming (about 13 %)

Consider the following program written in X86prime-assembler.

```
.L0:
    subq $8, %rsp
    movq %r11, (%rsp)
    cbe $0, %rsi, .L13
    leaq (%rdx), %r11
    movq $0, %r10
    movq %r10, (%r11)
    leaq (%rcx), %r11
    movq $0, %r10
    movq %r10, (%r11)
    cbae $1, %rsi, .L13
    movq $1, %eax
    jmp .L2
.L1:
    addq $1, %rax
    cbe %rax, %rsi, .L13
.L2:
    leaq (%rdi, %rax, 8), %r11
    movq (%r11), %r8
    movq (%rdx), %r10
    cbae %r10, %r8, .L1
    movq %r8, (%rdx)
    movq %rax, (%rcx)
    jmp .L1
.L13:
    movq (%rsp), %r11
    addq $8, %rsp
    ret %r11
```

Assembler programming, 1.1.1: Describe the details of the program. This includes (but is not limited to):

- Identify the instructions that implements the call convention; makes it possible to call the piece of code as a function.
- Identify the control structures of the program (e.g. loops, conditionals, and function calls).
- Identify registers which must have specific values at call time, for the function to have a specific purpose.
- Specify for each used register, the C type that best reflects the use of the register. (If a register is used in more than one way, specify this.)

Assembler programming, 1.1.2: Rewrite the above X86prime-assembler program to a C program. The resulting program must not have a goto-style and minor syntactical mistakes are acceptable.

Assembler programming, 1.1.3: Descripe shortly the functionality of the program.

Exam number: _____

1.2 Performance / Pipeline (about 10 %)

Consider the following execution graph (afviklingsplot) that illustrates the execution of a sequence of instructions on a 3-step pipeline machine, similar to the one used in A2.

```
.L3:
    movq (%r14), %rax           FXM
    cbe %rax, %rsi, .L1          FXXM            1)
.L4:
    leaq (%rdi, %rax, 8), %r11    FFXM           2)
    movq (%r11), %r8                FXM
    movq (%rdx), %r10                FXM
    cbae %r10, %r8, .L3               FXXM       3)
    movq %r8, (%rdx)                   FFXM      4)
    movq %rax, (%rcx)                     FXM
    jmp .L3                                FXM
.L3:
    addq $1, %rax                             FXM
```

Performance / Pipeline, 1.2.1: The numbes 1) to 4) indicates the four situations, where an instruction is in the same pipeline step for more than one machine cycle.

For each of the four cases describe why this is the case.

Performance / Pipeline, 1.2.2: In the note on execution graphs another "simple" pipeline with 5 steps is described; see https://x86prime.github.io/afviklingsplot/pipeline/ for more details.

Make an execution diagram, showing the execution of above code on this machine. Describe the interesting parts; e.g. when and why you stall the pipeline.

Performance / Pipeline, 1.2.3: A 5-step pipeline is expected to perform less work in each step and thus have a shorter longest signal path. Assume therefore that the 5-step pipeline is 25 % faster (in the sense that it's clock frequency is 25 % faster) than the 3-step pipeline.

Which of the two pipelines executes the program sequence fastest? Why is this?

1.3 Data Cache (about 10 %)

Data Cache, 1.3.1: You can choose between a "direct mapped" and a 4-way associative cache. The two caches have same total size and access time.

Which type of cache will you choose and why?

Data Cache, 1.3.2: Explain shortly the difference between "write-back" and "write-through" caching.

Data Cache, 1.3.3: In the following we examine a 4-way set-associative data cache with 16 cache-blocks of 16 bytes each.

Indicate for each of the references in the following stream, if the cache access is a miss or a hit. Addresses are given in heximal notation, and touches a single byte. Assume the cache is cold on entry.

`0x000, 0x001, 0x01F, 0x102, 0x203, 0x304, 0x10C, 0x705, 0x002, 0x10F, 0x210, 0x010.`

Data Cache, 1.3.4: Now assume that the cache is only 2-way set-associative, but the total size is the same as the one above.

Again indicate for each of the references in the following stream, if the cache access is a miss or a hit. Addresses are given in heximal notation, and touches a single byte. Assume the cache is cold on entry.

`0x000, 0x001, 0x01F, 0x102, 0x203, 0x304, 0x10C, 0x705, 0x002, 0x10F, 0x210, 0x010.`

Data Cache, 1.3.5: Explain the difference between the 2 sequences of events ? Why do you see these differences?

Exam number: _____

2 Operating Systems (about 33 %)

2.1 Multiple Choice Questions (about 6 %)

In each of the following questions, you may put one or more answers. Use the lines to argue for your choices.

Multiple Choice Questions, 2.1.1: Which of the following are states that a Unix process can be in?

☐ **a)** Crashed

☐ **b)** Running

☐ **c)** Reaped

☐ **d)** Stopped

☐ **e)** Waiting

☐ **f)** Zombie

Multiple Choice Questions, 2.1.2: Why is virtual memory useful?

☐ **a)** To download RAM from the Internet

☐ **b)** So processes can use more memory than will fit in RAM

☐ **c)** Protecting a process's memory from being accessed by the kernel

☐ **d)** Protect a process's memory from being accessed by another process

☐ **e)** More efficient use of RAM

☐ **f)** Because otherwise `malloc()` could not be implemented.

Multiple Choice Questions, 2.1.3: What is the purpose of the dirty bit in page table entries?

☐ **a)** Avoiding unneeded disk writes

☐ **b)** Marking the page as read-only.

☐ **c)** Maintains LRU information for page replacement stragies.

☐ **d)** Whether the entry is valid.

2.2 Short Questions (about 12 %)

Short Questions, 2.2.1: Does the following program contain race conditions? Why or why not?

```c
#include <unistd.h>
#include <stdio.h>

int x = 0;

void* worker(void* p) {
  for (int i = 0; i < 1000; i++)
    x++;
  return NULL;
}

int main() {
  pthread_t t;
  pthread_create(&t, NULL, thread, NULL);
  pthread_join(t);
  pthread_create(&t, NULL, thread, NULL);
  pthread_join(t);
  printf("%d\n", x);
}
```

Exam number: _____

Short Questions, 2.2.2: Consider a system with the following properties:

- Memory is byte-addressed.
- Virtual addresses are 14 bits wide.
- Physical addresses are 13 bits wide.
- The page size is 16 bytes.
- The TLB is 3-way set associative with 8 sets and 24 total entries. Its initial contents are:

Set	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
0	0F	10	1	02	00	1	09	00	0
1	20	10	0	11	15	1	1F	2F	1
2	12	15	1	00	12	0	19	0A	0
3	4F	10	1	12	00	0	19	00	0
4	21	34	1	11	00	0	30	0A	1
5	30	10	0	41	15	1	2F	3F	1
6	22	15	1	10	12	0	29	FA	0
7	01	00	0	31	34	1	20	00	1

- The page table contains 12 PTEs:

VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid	VPN	PPN	Valid
02	33	1	0C	0D	0	08	17	1	13	11	1
32	33	1	01	02	0	41	01	1	04	01	1
10	21	0	00	00	1	A2	32	0	03	43	1

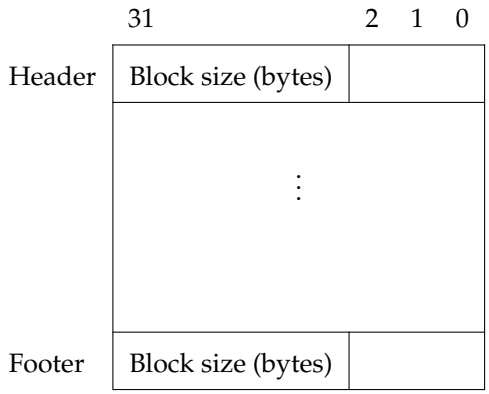
Note that all addresses are given in hexadecimal. In the following questions, you are asked, for various virtual addresses, to show the translation from virtual to physical addresses in the memory system just described. *Hint: there is one TLB hit, one page table hit, and one page fault (not necessarily in that order). This should help you double-check your work.*

Also answer the following: Describe shortly how you calculated your answers.

Exam number: _____

2.3 Long Questions (about 15 %)

Long Questions, 2.3.1: Consider an allocator that uses an implicit free list. The layout of each allocated and free memory block is as follows, with one 32-bit word per row:



Each memory block, either allocated or free, has a size that is a multiple of eight bytes, rounding up allocations if necessary. Thus, only the 29 higher order bits in the header and footer are needed to record block size, which includes the header and footer. The usage of the remaining 3 lower order bits is as follows:

- `bit 0` indicates the use of the current block: 1 for allocated, 0 for free.
- `bit 1` indicates the use of the previous adjacent block: 1 for allocated, 0 for free.
- `bit 2` is unused and is always set to be 0.

Important: We must *never* create blocks with zero payload (i.e. we must *never* create blocks with size 8).

Given the heap shown on the left, show the new heap contents after *consecutive* calls to

1. `free(0x500c010)`.
2. `realloc(0x500c000, 12)`. Assume the the return value is `0x500c000`, meaning that the existing allocation is resized.

Your answers should be given as hex values. Note that the address grows from bottom up. Assume that the allocator uses immediate coalescing, that is, adjacent free blocks are merged immediately each time a block is freed.

Address	Original value	After `free`	After `realloc`
0x500c028	0x00000013		
0x500c024	0x500c611c	0x500c611c	0x500c611c
0x500c020	0x500c512c	0x500c512c	0x500c512c
0x500c01c	0x00000013		
0x500c018	0x00000013		
0x500c014	0x500c511c	0x500c511c	0x500c511c
0x500c010	0x500c601c	0x500c601c	0x500c601c
0x500c00c	0x00000013		
0x500c008	0x00000013		
0x500c004	0x500c601c	0x500c601c	0x500c601c
0x500c000	0x500c511c	0x500c511c	0x500c511c
0x500bffc	0x00000013		

Also answer the following: Does the resulting heap exhibit internal or external fragmentation? Explain your answer.

Long Questions, 2.3.2: Explain, in pseudocode or prose, how semaphores can be implemented in terms of mutexes and integer variables. Show how to initialise a semaphore with initial value v, how to implement the P operation, and how to implement the V operation. Could your implementation be made more efficient by also using condition variables?

Exam number: _____

3 Computer Networks (about 34 %)

3.1 Application layer (about 6 %)

Application layer, 3.1.1: Suppose you are starting a new company and want to host the web-page of the company. Discuss the difference between hosting the web-page using a content distribution network like Akamai in comparison to hosting it on peer-to-peer networks like BitTorrent.

Application layer, 3.1.2: What role does DNS play for content distribution networks and how is it used?

3.2 Reliable Data Transfer (about 8 %)

Reliable Data Transfer, 3.2.1: How does flow control in TCP help with congestion control? Why is flow control not enough to deal with congestion control and conversely congestion control not enough to deal with flow control?

Reliable Data Transfer, 3.2.2: Suppose there are k TCP connection ongoing over a shared link with bandwidth R. In addition to the k TCP connections, another TCP connection is initiated over this shared link. What is the bandwidth available to the new TCP connection and why?

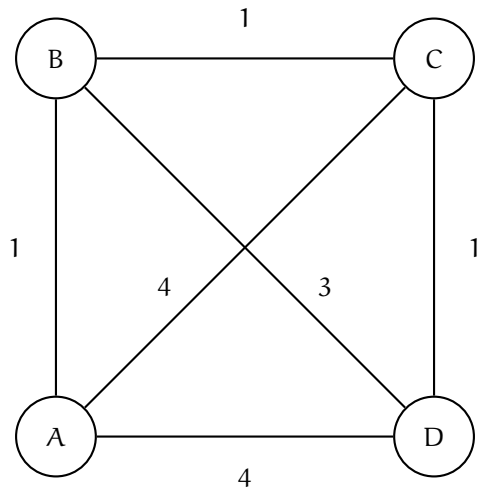
Reliable Data Transfer, 3.2.3: If instead of TCP, UDP had been used for the k connections, how much bandwidth would be available to the new UDP connection and why?

3.3 Network Layer (about 10 %)

Network Layer, 3.3.1: Why is an IP address considered hierarchical? What problems do hierarchical IP addresses solve and how?

Network Layer, 3.3.2: What is the purpose of MAC addresses? Why are MAC addresses not hierarchical?

Consider the network topology outlined in the graph below and suppose we have used the distance vector routing algorithm to calculate shortest paths.



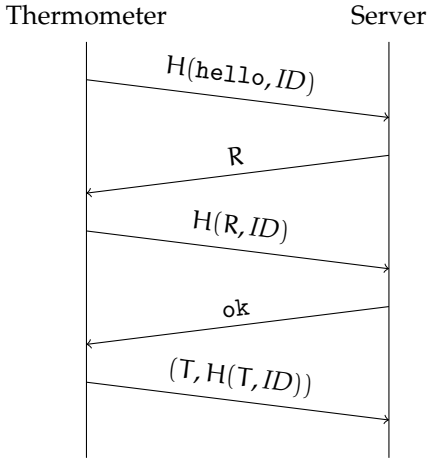
Network Layer, 3.3.3: Now the cost of the edge between A and D changes to 1, would a re-run of the algorithm converge? If yes, how many iterations of the algorithm would be run before convergence and why? If no, explain your reasoning. *(Hint: it is recommended that you run the distance vector algorithm on paper to explore the answer.)*

Network Layer, 3.3.4: Why can there be a problem with count-to-infinity and when can is occur? How could it happen in the graph given above.

3.4 Network Security (about 10 %)

You have been hired at a company that makes home-IoT devices and have been tasked to evaluate their device that at intervals registers the room temperature and reports this to a central server. This is used as part of their smart-home setup to control automatic thermometers and ventilation.

The design is based on the following communication diagram, that communicates using TCP. H is a cryptographic hash function, ID is a unique identifier for each thermometer that only the thermometer and server knows.



- The thermometer sends a hashed hello message with the ID. The values are hashed, so the ID remains secret.
- The server finds the thermometer ID by looking up $H(\texttt{hello}, ID)$ in its database after which it replies with a nonce, R.
- The thermometer returns the hashed value of the nonce with the ID.
- The server checks that $H(R, ID)$ is correct and replies with an ok message.
- The thermometer sends the temperature, T, with the hashed ID and T.
- The server checks that $H(T, ID)$ matches with T and registers the temperature.

Network Security, 3.4.1: Is this system vulnerable to a *replay-attack*? Explain your answer. If the system is secure, point to what makes it secure. If the system is vulnerable, explain what makes is wrong and how an attack can be performed.

Network Security, 3.4.2: Is this system vulnerable to a *man-in-the-middle*? Explain your answer. If the system is secure, point to what makes it secure. If the system is vulnerable, explain what makes is wrong and how an attack can be performed.

Network Security, 3.4.3: The protocol does not use encryption at all. How can you use encryption to make the protocol more secure? Explain which kind of encryption you would use and in which stage of the protocol you use it.