CPSC 261

Sample Midterm Exam Questions

Note: the questions in this document do not constitute an actual midterm exam. However, the questions are representative of the kinds of questions that could be asked. Please keep in mind that this set of questions does not exhaust all the possibilities and therefore should not be used as your primary source of study material.

Question 1. Performance terminology

Define each of the following terms:

1. Latency
2. Throughput
3. Capacity
4. Utilization
5. Batching
6. Dallying
7. Speculation

Question 2. Cache terminology

Define each of the following terms:

1. Cache block
2. Cache line
3. Dirty bit
4. Write-back cache
5. Write-through cache
6. Write-allocate
7. No-write-allocate
8. Direct-mapped cache
9. Set associative cache
10. Associative cache
11. Cache block size
12. Tag (in the context of a cache)
13. Offset (in the context of a cache)

Question 3. Short answer

1. Why is consistency an important property of a memory abstraction?
2. What is the value of abstraction in systems design?
3. What does incommensurate scaling mean?
4. What does emergent properties mean?
5. What does propagation of effects mean?
6. In systems design, what does a trade-off refer to?
7. Name and describe three tools or ideas that can be used to deal with the complexity of a system.
8. What is the principle of iterative design in computer systems? Why doesn't it apply to most non-computer systems?

Question 4. C

If I have:

```
long x = 5, *p = &x, **pp = &p;
```

What is the value of:

1. `pp + 1`
2. `*pp + 1`

3. `**pp + 1`

What high level function does the following expression compute?

```
( a < b ? b : a )
```

What does the following C statement do?

```
*p++ = *q++;
```

Draw a diagram that describes the memory that results when the following code executes:

```
long x[4];
x[0] = 0xfeed;
x[1] = 0xdead;
x[2] = 0xbeef;
x[3] = 0xbead;
long *y[2];
y[0] = &x[0];
y[1] = &x[2];
long *p = y[0] + 1;
long *q = *(y + 1) + 1;
```

Your diagram should include a depiction of the sizes and values of the variables x, y, p, and q.

Question 5. Cache Organization

1. Describe how a direct mapped cache determines whether a particular block is in the cache
2. Describe how a 4-way set-associative cache determines whether a particular block is in the cache.

A cache can be characterized by 4 numbers:

1. S - The size of the cache in bytes
2. B - The size of a cache block
3. N – the number of lines in a set
4. W – the number of bits in an address

For each of the following cache characterizations determine the other asked for quantities:

Cache 1:

1. S = 4KB
2. B = 16B
3. N = 1
4. W = 32

Determine these quantities and explain briefly how you did so:

1. How many lines are in the cache?
2. How many sets are in the cache?
3. How many bits are used as the offset within a cache block?
4. How many bits are used as the tag within a cache line?

Cache 2:

1. S = 16KB
2. B = 8B

3. N = 8
4. W = 32

Determine these quantities and explain briefly how you did so:

1. How many lines are in the cache?
2. How many sets are in the cache?
3. How many bits are used as the offset within a cache block?
4. How many bits are used as the tag within a cache line?

Question 6. Cache Simulation

Imagine a processor with:

- 12 bit addresses
- a byte-addressable memory
- memory accesses of a single byte at a time

Attached to a 16 byte, 2-way set-associative cache with 2 byte blocks that uses Least Recently Used as the eviction algorithm.

Consider the following sequence of memory accesses (all in hex):

000 001 002 004 010 000 020 001 003 00a 00b 003 013

Fill in the following table with the tags of the blocks that are held in the cache after each memory access. The columns are labeled with the address of each access from the above list; the rows labeled SiLj correspond to the sets and lines in the cache (S0L0 is set 0 line 0, S2L1 is set 2 line 1). When both lines of a set are vacant, the block will be added to line0.

For example, the first access to location 0 will be in set 0, and since both lines in set 0 are vacant, it will be placed in line 0. The block at address 0 is not in the cache so this access is a Miss. It will be brought into the cache, and the tag of location 0, which is 0, will be stored in the table as shown. For each access that hits in the cache, put a * in the row in which it hits. You do the rest of the accesses.

	000	001	002	004	010	000	020	001	003	00a	00b	003	013
S0L0	000												
S0L1													
S1L0													
S1L1													
S2L0													
S2L1													
S3L0													
S3L1													
Hit or Miss	M												

What is a conflict miss?

Circle all of the misses in the above table that are conflict misses.

Question 7. Assembly code

Here are three C functions:

```
int fun1(int a)
{
    return a * 30;
}
int fun2(int a)
{
    return a * 34;
}
int fun3(int a)
{
    return a * 18;
}
```

And three assembly functions:

```
fc:
     leaq (%rdi,%rdi,8), %rax
     addq %rax, %rax
     ret
fa:
     leaq (%rdi,%rdi), %rdx
     movq %rdi, %rax
     salq $5, %rax
     subq %rdx, %rax
     ret
fb:
     movq %rdi, %rax
     salq $5, %rax
     leaq (%rax,%rdi,2), %rax
     ret
```

Which assembly function corresponds to each C function? Explain your answer.

Notes:

1. The leaq instruction computes the address of its first operand and puts this address in the register given by the second argument.
2. The salq instruction shifts its second argument left by the number of bits given in the first argument, shifting in 0 bits in the low order bits.

Question 8. Inverse compilers

Here is the assembly language that results from compiling a C function. I want you to give me the C function. Remember that the first 2 arguments in x86_64 are in the %rdi and %rsi registers, respectively. The more high level, or natural, your C function is, the higher your grade will be.

```
     .globl  mystery
mystery:
        movq    $0, %rax
        testq   %rsi, %rsi
        jle     .L2
        movq    $1, %rcx
        movq    $0, %rdx
```

```
.L5:
        testq   %rcx, %rcx
        je      .L3
        addq    (%rdi,%rdx,8), %rax
        jmp     .L4
.L3:
        subq    (%rdi,%rdx,8), %rax
.L4:
        xorq    $1, %rcx
        addq    $1, %rdx
        cmpq    %rsi, %rdx
        jne     .L5
.L2:
        ret
```

Question 9. Linux shells

In the context of the Linux shell (the tcsh in particular if it matters) what does each of the following refer to:

1. Input redirection
2. Output redirection
3. Pipes
4. Variables

Give a sequence of Linux commands that compare the output of a program `p_a` with the output of a program `p_b`.

Question 10. Pipelined processors

Use what you know about pipelined processors to explain why:

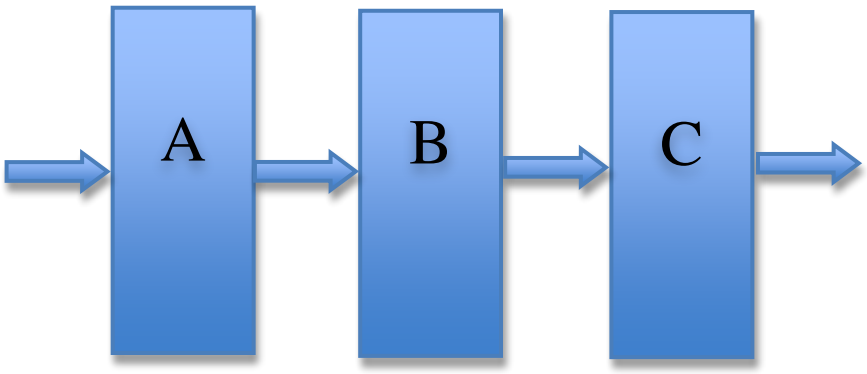
1. branches cause slow execution
2. dependencies between instructions cause slow execution
3. a pipeline with more stages is capable of executing more instructions per second than one with fewer stages (under the right conditions)

In the "classic" 5-stage RISC pipeline of the mid 1980s, the 5 stages are labeled Fetch, Decode, Execute, Memory, and Writeback. If the result of a conditional branch is not known until the end of the Memory stage, and the processor does not do any branch prediction, how many cycles are "wasted" by the pipeline each time a branch instruction enters the pipeline? Explain your answer.

If the 5-stage processor described above implements branch prediction, then in the cycle after a branch instruction enters the pipeline an instruction from the predicted branch target can enter the pipeline. In the case where the branch prediction was correct, no cycles are wasted. If branch prediction is incorrect, then the number of cycles wasted is as in the previous question. If branch prediction is right 3/4 of the time, then how many cycles, on average, are wasted for each branch? Explain your answer.

Question 11. Pipelined servers

Suppose that there is a service that is implemented with three modules: A, B, and C, organized as a pipeline as shown below. Requests are executed by the modules as shown by the arrows.



Suppose that all requests are independent. The processing time of module A is 10 msec, of module B is 20 msec, and of module C is 5 msec. Assume that there is a constant stream of requests flowing into module A.

1. What is the latency of this service? Explain.
2. What is the throughput of this service? Explain.
3. If you make module C twice as fast, how does this affect the latency and throughput of the service? Explain.
4. If in the original pipeline you split module B into two modules, B1 and B2 both requiring 10 msec of processing time and organized as a pipeline, how does this affect the latency and the throughput of the service? Explain.
5. If in the original pipeline you replicate module B, creating two copies of it that can execute requests concurrently, how does this affect the latency and the throughput of the service? Explain.
6. If in the original pipeline you replicate module B, creating **four** copies of it that can execute requests concurrently, how does this affect the latency and the throughput of the service? Explain.

Question 12. Amdahl's law

1. State Amdahl's law.
2. Use Amdahl's law to compute the speedup accomplished if you take a module that is currently consuming 20% of the time in a system and speed it up by a factor of 2. Explain your answer or show your work.
3. Use Amdahl's law to compute the speedup of the memory system accomplished by adding a cache to a system. A hit in the cache costs 5 cycles, a miss in the cache costs 100 cycles, and the hit rate in the cache is 90%. Explain your answer or show your work.
4. Repeat question 3 with a hit rate of 99%. Explain your answer or show your work.

Question 13. Locality

1. What is locality?
2. What is temporal locality?

3. What is spatial locality?
4. How does a cache take advantage of temporal locality?
5. How does a cache take advantage of spatial locality?
6. How can the parameters of a cache (block size, set size) be modified without making the cache bigger to take better advantage of temporal locality in a program? Explain.
7. How can the parameters of a cache (block size, set size) be modified without making the cache bigger to take better advantage of spatial locality in a program? Explain.