CS3330 Exam 3 – Fall 2014

Name: _____

Directions: Put the letter of your selection or the short answer requested in the box. Longer answers may use the space to the left of the box too. Write clearly: if we are unsure what you wrote you will get a zero on that problem.

If you do not sign the pledge on the last page you will get a zero on the entire quiz.

.....

Question 1: Suppose someone made a radically different architecture that was able to use something like quantum entanglement so that an unlimited number of instructions can be evaluated in parallel, possibly including many instructions passing through the same pipeline stage at once.

Which of the following reasons for bubbling or stalling a pipeline register will still apply?

A because of a branch misprediction
B because of a load-use dependency
C because a later stage in the pipeline bubbled or stalled
D all of the above still apply even if the CPU has infinite parallelism
E none of the above apply if the CPU has infinite parallelism

Answer: *B*

Question 2: Traps are unlike the various non-trap kinds of exceptions in that

A the are caused by a particular instruction executing
B the exception table is not involved in their resolution
C the exception table is involved in their resolution
D the user code that was running during when an interrupt occurred is always resumed after the trap is handled
E the user code that was running during when an interrupt occurred is never resumed after the trap is handled
F they are not caused by a particular instruction executing
G none of the above distinguishes traps from non-traps

Answer: *G*

Question 3: Variables exist in source code, but in assembly they are not present, having been replaced by memory addresses by the compiler. Labels are present in assembly, but not in an executing binary, having been replaced by addresses by the assembler. Labels are (generally) not present in source code; which of the following source code snippets cause the compiler to generate labels? Choose all that apply.

A `if (b) f();`
B `int x` as a parameter declaration
C `int x = 3;` as a global declaration
D `void f() { g(); }`
E `int x = 3;` as a local declaration

Answer: *if, void, and global*

Question 4:

What is decimal 42 in hexidecimal? (answer with just the hex digits, no leading 0x)

Answer: *2a*

Question 5:
Rank the following in order of significance when writing code that deals with large arrays:

- **A**lgorithmic efficiency (i.e., big-O),
- **C**omputational efficiency (e.g., loop unrolling, inlining, etc),
- cache **L**ocality.

Answer: *ALC*

List three letters with the most significant impact first (e.g., alphabetic order "ACL" would suggest algorithmic efficiency is most important and cache locality least important).

Question 6: Your code is accessing memory (assume it is non-cached, physical memory). Which of the following are involved in the access? Select all that apply (note: some terms are nonsense).

A I/O bus
B memory bus
C system interface
D cpu bridge
E I/O bridge
F memory bridge
G system bridge
H cpu bus
I system bus
J bus interface

Answer: *J*

Question 7: Code such as `*NULL` and `*(NULL+15)` will generate page faults because

A the compiler notices the `NULL` and inserts `int $14` (where 14 is the page fault exception number)
B a page fault will be generated, but not for any of the reasons above
C the small-address physical addresses are not mapped to by any virtual page in the page table
D the small-address virtual addresses are not mapped to any physical page in the page table
E none of the above because a page fault won't be generated

Answer: *D*

Question 8: Consider the following code:

```
int compositeness(int x) {
     int i;
     int ans = ((x % 2) == 0);
     for(i = 3; i < sqrt(x); i += 2)
         if (isPrime(i))
             if ((x % i) == 0)
                 ans += 1;
     return ans;
 }
```

Assume the compiler is performing no optimizations and that `isPrime` is the exact code presented in another question on this exam. Which of the following strategies would improve the speed of this code the most?

A swap the order of the if statements
B change the nested if statements to a single if with a `&&`
C add a second `ans` accumulator
D inline the `isPrime` call

Answer: *A*

Question 9: `a` is the 4-bit value 0011 and `b` is the 4-bit value 0110. What is `a - b`?

A 0011
B 0101
C 1101
D 1001
E 0010
F 0110
G 0111

Answer: *C*

Question 10: Jump tables, such as are used for switch statements, require that the new PC be determined by a memory read. Suppose we added an assembly instruction `memjump` accepting a single memory address as its parameter, like `memjump (%eax)`. In the five-stage example pipeline with data forwarding that we used this semester (F D E M W), for how many cycles will we need to bubble and/or stall stage F after executing a `memjump`?

A 1
B 4
C 5
D 3
E 2
F 0

Answer: *D*

Question 11: Good software design includes writing procedures for code you might otherwise repeat in-line. Pulling code into procedures can help some branch predictors; how else can it *improve* your program's performance and/or your compiler's ability to optimize your code?

A more opportunities for loop unrolling
B less chance of compiler having to worry about aliasing and side effects
C more opportunities for pipeline-level parallelism
D better instruction cache hit rate

Answer: *D*

Question 12: Consider the following code:

```
int * makeArray(int x) {
     int *ans = malloc(x*sizeof(int));
     for(i = 1; i <= x; i += 1)
         ans[i] = ans[i-1] * 2;
     return ans;
 }
```

Which of the following memory errors does it have? Select all that apply.

A confusing pointer and value sizes
B reading uninitialized memory
C off-by-one memory access error
D dereferencing a value as if it were a pointer
E returning references to nonexistent values

Answer: *unitialized and off-by-one*

Question 13: Core i7 uses a 4-level page table hierarchy with 4K pages for a total virtual address size of 48 bits. If we tweaked it to use 8K pages and a 3-level page table hierarchy, how large would virtual addresses be?

A 49 or 50 bits
B more than 50 bits
C 48 bits
D 42 or 43 bits
E less than 40 bits
F 40 or 41 bits
G 46 or 47 bits
H 44 or 45 bits

Answer: *D*

Question 14: Suppose page table entries are 8 bytes long and virtual addresses are 32 bytes long. If each page table is to fit on a single page, what is the smallest pages can be if we have a 2-level page table hierarchy?

A 256 B
B 1 KB
C 4 KB
D 512 B
E 2 KB
F smaller than any of the above options
G bigger than any of the above options

Answer: *G*

Question 15: Which one of the following optimizations does not provide any speed benefit if memory access is as fast as register access?

A loop unrolling
B invocation inlining
C reassociation
D storing reused computed values in variables instead of recomputing them
E using more efficient algorithms
F all of the above provide speed benefits for fast-memory processors

Answer: *F*

Question 16: We say that pipelining generally increases latency. That statement means (select all that are part of the definition of "increases latency"):

A each instruction takes less time to complete
B each instruction takes more time to complete
C entire programs take more time to complete
D entire programs take less time to complete

Answer: *B*

Question 17: Good software design includes writing procedures for code you might otherwise repeat in-line. Pulling code into procedures involves call/return overhead; how else can it *hurt* your program's performance and/or your compiler's ability to optimize your code?

A fewer opportunities for pipeline-level parallelism
B worse instruction cache hit rate
C more chance of compiler having to worry about aliasing and side effects
D fewer opportunities for loop unrolling

Answer: *C*

Question 18: `a` is the 4-bit value 0011 and `b` is the 4-bit value 0110. What is `a ^ b`?

A 1101
B 0011
C 1001
D 0111
E 0110
F 0101
G 0010

Answer: *F*

Question 19: You have code that is running slowly. You profile it and find the slow part is transposing 10,000-by-10,000 element matrices of `double` values. You want to buy hardware to speed it up. Assuming you are willing to optimize your code for the new hardware, you should probably

A upgrade from a 2.4GHz to a 3.2GHz processor clock speed
B upgrade from 2GB RAM to 32GB RAM
C upgrade from a two-level cache storing 1MB in the L2 to a four-level cache storing 256MB in the L4
D None of the above would make it much faster

Answer: *C*

Question 20: Consider the following code:

```
void f(int** a, int**b, unsigned n, unsigned m) {
     for(i = 0; i < n; i += 1)
         for(j = 0; j < m; j += 1)
             a[j][i] = b[(j*101)%m][(i*29)%n];
 }
```

Which of the following changes would make the largest positive impact on its cache locality?

A Put the for-i loop inside of the for-j loop instead of the other way around
B Change `int**` to `int*`
C Block the accesses by nesting three or four for loops instead of just two
D Store the result of `(i*101)%n` outside the inner loop

Answer: *A*

Question 21: The condition codes are stored in registers; why do we not call those registers "program registers"?

A you cannot write code that reveals their contents
B they aren't big enough (too few bits)
C they cannot be bubbled or stalled
D you cannot write code that uses them as operands
E you cannot write code that changes their contents

Answer: D

Question 22:
Consider the following pipeline diagram:

A	F	D	E	M	W			
B		F	D	E	E	M	W	
C			F	D	D	E	M	W

This diagram suggests that *C*'s ___ phase depends on ___'s ___ phase? Answer with three letters (the second will be either *A* or *B*); if it suggests *C* has multiple dependencies, answer "multiple;" if it does not suggest *C* has any dependency, answer "none".

Answer: none

Question 23: Compare the size of `int` and `int*`.

A `int` has fewer bits
B `int*` has fewer bits
C they have the same number of bits
D Which one of the above depends on the computer

Answer: D

Question 24:
Consider the 5-stage pipeline (F D E M W); assume each phase has a pipeline register feeding it (e.g., five pipeline registers, the first being before F and the last being between M and W).
If during the M stage we discover that all the instructions we've executed since the instruction in M were based on a mis-predicted branch, how many of the registers will be given normal, stall, and bubble signals? Answer as 3 digits (normal stall bubble) that sum to 5 (e.g., if 3 are bubbled and 2 stalled answer "0 2 3").

Answer: 2 0 3

Question 25: The bias of any IEEE-style floating point number is $2^{e-1} - 1$, where e is the number of exponent bits. Suppose 010010 is a six-bit IEEE-style floating point number that represents the decimal value 2.5. How many exponent bits does the number have?

A 1
B 5
C 4
D 3
E 2

Answer: E

Question 26:
An 5-bit address is sent to a cache with 4 sets of 2 lines; each block contains 2 1-byte words. How many bits long are the tag, set index, and block offset (including both word-of-block and byte-of-word)? Answer with three numbers (e.g., if the tag is 0 bits, the set index 2 bits, and the block offset 3 bits, answer "0 2 3")

Answer: 2 2 1

Question 27: The exception table is

A a data structure describing which exception types the hardware should handle
B a data structure representing what caused the exception
C a list of addresses of exception handlers
D a list of the exceptions that your code has thrown
E none of the above

Answer: C

Question 28: The contents of how many program registers are modified by an assembly `push` operation?

A 0
B 2
C 1
D 3 or more
E it depends on that argument of the push

Answer: C

Question 29: What kind of locality does the following code benefit from?

```
void f(int* a, unsigned n, unsigned m) {
     for(i = 0; i+1 < n; i += 1)
         for(j = m-1; j >= 0; j -= 1)
             a[(m-1-j)*n + i] = 0;
 }
```

Assume that `n` and `m` are very large numbers.

A spatial locality
B temporal locality
C both of the above
D none of the above

Answer: D

Question 30: We say that pipelining generally increases throughput. That statement means (select all that are part of the definition of "increases throughput"):

A entire programs take more time to complete
B entire programs take less time to complete
C each instruction takes less time to complete
D each instruction takes more time to complete

Answer: B

Question 31: We discussed that idea that page table maps a (process id, virtual address) pair into a physical address; however, nothing in the MMU has a process id. The OS and architecture still work together to give us per-process virtual addresses because

A There is a register set by the OS when it switches processes and used by the MMU at the beginning of the address translation process
B The MMU asks the OS which page tables to read during every address translation
C The OS sets the "invalid" bit of all the old processes' page table entries when it switches processes
D The OS changes the contents of all the page tables when it switches processes
E There is a register set by the OS when it switches processes and used by the MMU at the end of the address translation process
F The MMU asks the OS which page tables to read if the TLB has a miss (but not if it has a hit)

Answer: *A*

Question 32: Suppose a cache has access to the set index and block offset before it has access to the tag. This allows it to

A begin, but not complete, cache lookup before the tag is available
B complete cache lookup before the tag is available
C neither begin nor complete cache lookup before the tag is available
D which of the above depends on if the cache is associative or not

Answer: *A*

Question 33:

What is -5 as a six-bit two's-compliment number? Answer in binary (exactly six bits, each either 0 or 1)

Answer: *111011*

Question 34:
Consider the following pipeline diagram:

A	F	D	E	M	W			
B		F	D	E	E	M	W	
C			F	D	D	E	M	W

This diagram suggests that *B*'s ___ phase depends on ___'s ___ phase? Answer with two letters (the second will be either *A* or *C*); if it suggests *B* has multiple dependencies, answer "multiple;" if it does not suggest *B* has any dependency, answer "none".

Answer: *E A M*

Question 35: A cache has a cold miss is when (pick the most specific true answer)

A the address being accessed was not in the cache
B no address with the same set index was previously accessed
C the address being accessed used to be in the cache but is not anymore
D the cache line being accessed was previously invalid
E no address with the same set index and tag was previously accessed
F no address with the same tag was previously accessed
G the exact address being fetched was never previously accessed

Answer: *E*

Question 36: Given that `a` and `b` are each either 0 or 1, which one of the following is *not* always true?

A `(a ^ b) == ((a - b) & 1)`
B `(a ^ b) == (a != b)`
C `(a ^ b) == ((a + b) >> 1)`
D `(a ^ b) == (a ? !b : b)`
E `(a ^ b) == (1 & (a + b))`
F all of the above are always true

Answer: C

Question 37:
Consider the 5-stage pipeline (F D E M W); assume each phase has a pipeline register feeding it (e.g., five pipeline registers, the first being before F and the last being between M and W).
If E needs another cycle to work but the other stages are ready to continue, how many of the registers will be given normal, stall, and bubble signals? Answer as 3 digits (normal stall bubble) that sum to 5 (e.g., if 3 are bubbled and 2 stalled answer "0 2 3").

Answer: 1 3 1

Question 38: The MMU's physical address space is determined by which two of the following? Your answer should be two letters between A and E.

A the size of memory pages
B the contents of page table entries
C size of the memory chip in your computer
D size of the virtual addresses
E number of levels in the page table hierarchy

Answer: PTE contents + page size

Question 39: Consider the following code:

```
bool isPrime(int x) {
     int i;
     for(i = 3; i < sqrt(x); i += 2)
         if ((x % i) == 0)
             return false;
     return (x % 2) != 0;
 }
```

Assume the compiler is performing no optimizations. Which of the following strategies would improve the speed of this code the most in the case where it returns `true`?

A move the check for `(x % 2) != 0` to before the loop
B unroll the loop
C declare `int sx = (int)sqrt(x)` and change `i < sqrt(x)` to `i < sx`
D pull the `(x % i) == 0` into a function `isDivisibleBy`
E change `i < sqrt(x)` to `i*i < x`

Answer: C

Question 40: Suppose an array of `char` (8-bit values) {`0x12, 0x34, 0x56, 0x78`} is stored at address 0x24 of a little-endian computer. What byte is stored at address 0x26?

A 0x56
B 0x00
C 0x65
D A known value not listed here
E The byte at address 0x26 is not set by this array
F 0x34
G 0x43

Answer: *A*

Question 41: Virtual addresses and caches are similar in that both

A can impact the performance of memory reads
B translate one address into another address
C are managed by cooperative effort of the operating system and the hardware
D are capable of generating exceptions/faults/interrupts

Answer: *A*

Question 42: Which one of the following optimizations does not provide any speed benefit for a non-pipelined processor, such as SEQ?

A storing reused computed values in variables instead of recomputing them
B using more efficient algorithms
C loop unrolling
D invocation inlining
E reassociation
F all of the above provide speed benefits for non-pipelined processors

Answer: *E*

Question 43: If `x` is the most negative signed integer, what is `-x`?

A one
B negative one
C the most negative signed number
D the most positive signed number
E zero

Answer: *C*

Question 44: If a page fault is resolved by the OS loading a new page into memory, once the page fault handler is done and control returned to the user

A the user program throws a user-code Exception and resumes at the nearest `catch` block
B the user program runs the instruction after the memory access instruction
C the user program aborts
D the user program re-runs the memory access instruction
E none of the above

Answer: *D*

Question 45: Interrupts are unlike the various non-interrupt kinds of exceptions in that

A the user code that was running during when an interrupt occurred is always resumed after the interrupt is handled
B they are not caused by a particular instruction executing
C the exception table is not involved in their resolution
D the user code that was running during when an interrupt occurred is never resumed after the interrupt is handled
E the are caused by a particular instruction executing
F the exception table is involved in their resolution
G none of the above distinguishes interrupts from non-interrupts

Answer: B

Pledge:

On my honor as a student, I have neither given nor received aid on this exam.

_____

Your signature here