EECS 370

Fall 2003

Final Exam

December 15, 2003

Name:

Uniqname:

Honor pledge: I have neither given nor received aid on this examination, nor have I concealed violations of the honor code.

Signature:

Open book, open notes, calculators permitted. No laptops, cellphones, PDA's, etc. This exam has 13 questions, 11 pages, and 100 points. The last page has reference information on the LC-2 and on small and large numbers that you may find helpful.

1. *(3 points)* The primary advantage of multi-level page tables is that they:

   (a) use less memory
   (b) make TLB accesses faster
   (c) make page table accesses faster
   (d) allow the page tables to reside in virtual memory
   (e) reduce the degree of associativity needed for the TLB

2. *(3 points)* TLBs reduce the amount of time it takes to:

   (a) perform an address translation
   (b) evict a dirty block from the cache
   (c) load a page table entry into memory
   (d) switch to a new process address space
   (e) flush the pipeline after a mispredicted branch

3. *(3 points)* You recently added the ability to do memory-mapped I/O to an LC2K3 processor. Let's say you want to obtain data from the keyboard which can be accessed using address 0x8003. Which of the following instructions (written in machine code) would place the keyboard data into register 7? Assume this is the current state of the register file: r0=0, r1=1, r2=2, r3=3, r4=4, r5=5, r6=6, r7=7, pc=0x4000

   (a) `0x00A77FFF`
   (b) `0x00874002`
   (c) `0x00878003`
   (d) `0x00C78003`
   (e) Not possible - a change to the instruction set encoding is required

4. *(2 points)* No benefit to overall system performance can be achieved by increasing the speed of the I/O bus (between I/O devices and memory) beyond the speed of the CPU bus (between the processor and memory).

   TRUE   FALSE

5. *(4 points)* What is the maximum number of ROM storage bits required to implement a finite state machine with 3 input bits, 33 internal states, and 2 bits of output?

__________ bits

6. Given a 32-bit virtual address, a 32768 byte page size, and byte addressable memory:

(6A) *(4 points)* For a program that uses the entire address space, what is the total size of a single-level page table if all page table entries are 4 bytes in size?

__________ bytes

(6B) *(4 points)* If a two-level page table arrangement is used with 512 entries in the top level table, what is the total size of the entire page table? (For the same program as in part 6A, and all page table entries for all levels are 4 bytes in size.)

__________ bytes

7. Given a disk with the following characteristics:
   - Average wait time: 10 milliseconds
   - Average seek time: 30 milliseconds
   - Average rotational delay (or latency): 5 milliseconds
   - Sector size: 2 megabytes
   - Sectors per track: 50 (constant)
   - Tracks per disk: 100
   - Platters per disk: 8
   - Surfaces per disk: 15
   - Only one surface can be read at a time
   - Data is located sequentially on the disk

   (7A) *(3 points)* What is the average time to read 10 MB from disk?

   __________ milliseconds

   (7B) *(3 points)* What is the minimum time to read 10 MB from the disk?

   __________ milliseconds

8. Datapaths

(8A) *(3 points)*

In an implementation of a single-cycle datapath, the clock period is 10 ns. The critical path is comprised of many components of the datapath, including the ALU. To reduce costs, another implementation is done replacing the 2 nanosecond ALU with a 5 nanosecond one. If a program took 30 secs to run on the first implementation, how long will it take to run on the second implementation?

_____ seconds

(8B) *(3 points)*

In an implementation of a multi-cycle datapath, the clock period is 3 ns. The multi-cycle datapath has a separate state for performing the ALU operation; in this state the only active component is the ALU. To reduce costs, another implementation is done replacing the 2 nanosecond ALU with a cheaper 5 nanosecond one. If a program took 30 secs to run on the first implementation, how long will it take to run on the second implementation?

_____ seconds

9. Your processor implements the LC2K3 pipeline as discussed in class. In particular, register writes occur in the first half of the clock cycle while register reads occur in the second half of the clock cycle and data forwarding is used to resolve stalls when possible. Use this code sequence for the following questions.

```
add  3 4 5
add  5 6 7
add  6 5 1
lw   3 1 9
nand 1 5 2
add  2 3 4
sw   2 3 4
lw   4 4 6
sw   4 4 6
```

(9A) *(3 points)* How many cycles does it take to execute this code sequence?

(a) 13
(b) 14
(c) 15
(d) 16
(e) 17
(f) 18

(9B) *(3 points)* The first instruction is in the fetch stage in cycle 1. What instruction is in the decode stage in cycle 7?

(a) add 6 5 1
(b) lw 3 1 9
(c) nand 1 5 2
(d) add 2 3 4
(e) noop
(f) none of the above

(9C) *(3 points)* If the forwarding path from the MEM/WB pipeline register to the EX stage was removed, how many "additional" cycles would it take to execute this code sequence?

(a) 0
(b) 1
(c) 2
(d) 3
(e) 4
(f) 5

10. *(11 points)* The page table, TLB, and cache contents are shown below for a system that uses a byte addressable memory with a 256 byte page size. Virtual addresses are 17 bits and physical addresses are 12 bits. The system has a 64 byte 2-way set associative physically addressed cache with a 4 byte block size. It also has a 4-way set associative TLB with 16 entries.

Page Table

VPN	PPN	Valid	VPN	PPN	Valid
0x000	C	0	0x010	1	1
0x001	7	1	0x011	8	1
0x002	3	1	0x012	3	0
0x003	8	1	0x013	E	1
0x004	0	0	0x014	6	0
0x005	5	0	0x015	C	0
0x006	C	1	0x016	7	0
0x007	4	1	0x017	2	1
0x008	D	1	0x018	9	1
0x009	F	0	0x019	A	0
0x00A	3	1	0x01A	B	0
0x00B	0	1	0x01B	3	1
0x00C	0	0	0x01C	2	1
0x00D	F	1	0x01D	9	0
0x00E	4	0	0X01E	5	0
0x00F	7	1	0x01F	B	1

TLB

Index	Tag	PPn	Valid
0	0x55	6	0
	0x48	F	1
	0x00	C	0
	0x77	9	1
1	0x01	4	1
	0x32	A	1
	0x02	F	0
	0x73	0	1
2	0x02	3	1
	0x0F	B	0
	0x04	3	0
	0x26	C	0
3	0x00	8	1
	0x7A	2	1
	0x21	2	0
	0x17	E	0

2-Way Set Associative Physically Addressed Cache

Index	Tag	Valid	Byte 0	Byte 1	Byte 2	Byte 3	Tag	Valid	Byte 0	Byte 1	Byte 2	Byte 3
0	7A	1	09	EE	12	64	00	0	99	04	03	48
1	02	0	60	17	18	19	38	1	00	BC	0B	37
2	55	1	30	EB	C2	0D	0B	0	8F	E2	05	BD
3	07	1	03	04	05	06	5D	1	7A	08	03	22
4	12	0	06	78	07	C5	05	1	40	67	C2	3B
5	71	1	0B	78	07	C5	6E	0	B0	39	D3	F7
6	91	1	A0	B7	26	2D	F0	0	0C	71	40	10
7	46	0	B1	0A	32	0F	DE	1	12	C0	88	37

The processor reads one byte from virtual address 0x01FAD. Indicate what happens during the address translation process and the cache access by filling in the blanks provided. All numeric answers should be given in hexadecimal.

Virtual page number:	0x______	Physical address:	0x______
Page offset:	0x______	Cache block offset:	0x______
TLB index:	0x______	Cache index:	0x______
TLB tag:	0x______	Cache tag:	0x______
TLB hit (yes/no):	______	Page fault (yes/no):	______
Cache byte returned (or put **MISS** if appropriate):			0x______

11. The designers of the Foo-2 computer decide not to follow the IEEE 754 floating point standard. Instead, they modify it to have 1 sign bit, 9 exponent bits, and 22 significand bits. The fields are laid out in the same order as in the IEEE standard; first the sign, then the exponent, then the significand. The exponent bias used is 255. We will refer to this representation as "Foo-2" format. Except for the different field sizes, Foo-2 floating point works in the same way as standard IEEE floating point. One consequence of the different formats is that data cannot be correctly moved between a Foo-2 machine and a machine using the IEEE standard without appropriate conversion.

(11A) *(2 points)* All numbers that can be exactly represented in Foo-2 floating point can also be exactly represented in IEEE standard floating point.

TRUE FALSE

(11B) *(2 points)* All numbers that can be exactly represented in IEEE standard floating point can also be exactly represented in Foo-2 floating point.

TRUE FALSE

(11C) *(2 points)* Overflow may occur when converting a number from Foo-2 format to IEEE format.

TRUE FALSE

(11D) *(2 points)* Overflow may occur when converting a number from IEEE format to Foo-2 format.

TRUE FALSE

(11E) *(2 points)* Rounding may be necessary when converting a number from Foo-2 format to IEEE format.

TRUE FALSE

(11F) *(2 points)* Rounding may be necessary when converting a number from IEEE format to Foo-2 format.

TRUE FALSE

(11G) *(2 points)* Consider the total number of distinct values (real numbers) that can be represented exactly in each of the two formats. Which of the following three statements is most accurate:

(a) Foo-2 can represent many more values than IEEE

(b) IEEE can represent many more values than Foo-2

(c) Foo-2 and IEEE can represent about the same number of values

12. Answer the questions below for a processor with the following characteristics. Be sure to state your answer in the specific units given.

- 2.2 GHz clock speed
- Byte addressable memory
- Addresses are 40 bits
- Separate level 1 instruction and data caches
- Both level 1 caches have a 64 KB data size
- Both level 1 caches are 2 way set associative
- Combined (instruction and data) level 2 cache
- 512 kilobyte level 2 cache size
- Level 2 cache is 16 way set associative
- 64 byte cache block size (for caches at all levels)
- Write allocate, write back cache policies
- 4 kilobyte page size
- 4 way set associative TLB with 1024 sets
- IEEE floating point support (single precision only)

(12A) *(2 points)* How large is the block offset for the level-2 cache? ____________ bits

(12B) *(2 points)* How large is the set index for the level-2 cache? ____________ bits

(12C) *(2 points)* How large is the tag for the level-2 cache? ____________ bits

(12D) *(2 points)* How large is the block offset for the level-1 data cache? ____________ bits

(12E) *(2 points)* How large is the set index for the level-1 data cache? ____________ bits

(12F) *(2 points)* How large is the tag for the level-1 data cache? ____________ bits

(12G) *(2 points)* What is the smallest total program size (both instructions and data) that could cause **capacity** misses in the TLB? ____________ pages

(12H) *(2 points)* What is the smallest total program size that could cause **conflict** misses in the TLB? ____________ pages

(12I) *(2 points)* What is the smallest total program size that could cause **capacity** misses in the level 2 cache? ____________ pages

(12J) *(2 points)* What is the cycle time of this processor? ____________ picoseconds

(12K) *(3 points)* The fraction $-\frac{173}{512}$ is stored in a floating point register. What is the value in hex of this register? 0x____________________________

13. *(8 points)* Below is the definition of Ackerman's function, a recursive function of two variables m and n that grows extremely rapidly, along with an incomplete MIPS function to compute it. m is passed through $a0 and n through $a1. The return value is passed back in $v0. Fill in the missing parts of the function. You may assume that the input values for $m, n \geq 0$ (that is, you do not have to do any error checking). Note that the stack grows downward in MIPS. You may ONLY add code in the three specified places.

```
# Definition of  A(m, n):
#   when m = 0:            A(0, n) = n+1
#   when m > 0 and n = 0:  A(m, 0) = A(m-1, 1)
#   when m > 0 and n > 0:  A(m, n) = A(m-1, A(m, n-1))

ackerman:                            # push things on the stack
                                     # ADD YOUR CODE HERE





        beq $a0, $0, mzero           # case: m = 0
        beq $a1, $0, nzero           # case: n = 0

        addi $a1, $a1, -1            # $a1 = n - 1
        jal ackerman                 # call A(m, n-1)

                                     # prepare for calling A(m-1, A(m, n-1)
                                     # ADD YOUR CODE HERE





        jal ackerman                 # call A(m-1, A(m, n-1))
        j ret_ackerman               # clean up before returning

mzero:
        addi $v0, $a1, 1             # $v0 (the return value) = n + 1
        j ret_ackerman               # clean up before returning
nzero:
        addi $a0, $a0, -1            # $a0 = m - 1
        addi $a1, $0, 1              # $a1 = 1
        jal ackerman                 # call A(m-1, 1)

ret_ackerman:                        # pop things from stack and return
                                     # ADD YOUR CODE HERE
```

Reference Information

LC-2K3 quick summary:

The LC-2K3 is an 8-register, 32-bit computer. All addresses are word-addresses. The LC-2K3 has 65536 words of memory. By assembly-language convention, register 0 will always contain 0 (i.e. the machine will not enforce this, but no assembly-language program should ever change register 0 from its initial value of 0). There are 4 instruction formats. Bit 0 is the least-significant bit. Bits 31-25 are unused for all instructions, and should always be 0. Offsets are 16 bit signed two's complement values.

Bits	Field
24 - 22	opcode
21 - 19	reg A
18 - 16	reg B
15 - 0	offset
2 - 0	destReg

Instruction Mnemonic	Binary Opcode
add	000
nand	001
lw	010
sw	011
beq	100
jalr	101
halt	110
noop	111

Large quantities:

For this course, K, M, G, and T are used with their binary interpretations for data quantities:

kilo: 1 Kbyte = 2^{10} bytes = 1024 bytes
mega: 1 Mbyte = 2^{20} bytes = 1048576 bytes
giga: 1 Gbyte = 2^{30} bytes = 1073741824 bytes
tera: 1 Tbyte = 2^{40} bytes = 1099511627776 bytes

K, M, G, and T are used with their decimal interpretations for time and frequency quantities:

kilo: 1 KHz = 10^3 Hz = 1000 Hertz
mega: 1 MHz = 10^6 Hz = 1000000 Hertz
giga: 1 GHz = 10^9 Hz = 1000000000 Hertz
tera: 1 THz = 10^{12} Hz = 1000000000000 Hertz

Small times:

10^{-3} seconds = 1 millisecond
10^{-6} seconds = 1 microsecond
10^{-9} seconds = 1 nanosecond
10^{-12} seconds = 1 picosecond
10^{-15} seconds = 1 femtosecond