Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 460N Spring 2011
Y. N. Patt, Instructor
Faruk Guvenilir, Miad Hashemi, Yuhao Zhu, TAs
Exam 2
April 20, 2011

Name: Solution

Problem 1 (30 points):_____

Problem 2 (20 points):_____

Problem 3 (25 points):_____

Problem 4 (25 points):_____

Total (100 points):_____

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is recorded on each sheet of the exam.

Please sign the following. I have not given nor received any unauthorized help on this exam.

Signature: Solution

GOOD LUCK!

Name: Solution

Problem 1 (30 points)

~~*~~ Note: On the Exam Day, we clarified that each part takes 1 hour on a sequential machine

Part a (6 points): A problem takes 2 hours to solve on the computer. It consists of two parts, which must be done in sequence. The first part is 90% parallelizable, the second part is 60% parallelizable. Show your work below.

With infinite computing resources this problem would take how long?

30 min

$(1-.9)\cdot 60\text{min} = 6\text{ min} > 30\text{min}$

$(1-.6)\cdot 60\text{min} = 24\text{ min}$

This represents a maximum speed-up of what?

4 $\quad \frac{120\text{m}}{30\text{m}} = 4$

With 4 processors, the speed-up for this problem would be what?

120/52.5 $\quad \frac{60\times 2}{\frac{.9\times 60}{4} + (1-.9)\cdot 60 + \frac{.6\cdot 60}{4} + (1-.6)\cdot 60} = \frac{120}{13.5+6+9+24} = \frac{120}{52.5} = 16/7$

Part b (6 points): We have discussed in class various kinds of memory, one of which is a content-addressable memory. Two structures we have discussed in class actually use content addressable memory. For each case, identify the structure, and show all parts of an entry. Circle the element in each entry that is used for content addressing.

Name of structure: TLB

An entry in this structure: (Page Number) PTE

Name of structure: Tag Store

An entry in this structure: (Tag Bits), Valid Bit, Possibly Dirty/LRU Bits

Part c (6 points): Classical VLIW machines do not make effective use of their instruction caches. Why? Explain.

NOPs that must be inserted into the assembly code waste space.

Name: Solution

Problem 1 continued

Part d (6 points): We detect an interrupt or an exception; when does it get handled? We noted in class that interrupts are normally handled

When Convenient

whereas exceptions are normally handled

When Detected

However, as is often the case in computer architecture, "normally" is not the same as "always".

An example of an interrupt which is not handled at the time you stated above is Machine Check

An example of an exception which is not handled at the time you stated above is Overflow

Part e (6 points): Two device controllers A and B are on BR level #3. Device controller A is closer to the PAU. Suppose device controller A does not want the bus, but device controller B does. This is the highest priority bus request so the PAU grants the bus with BG #3. Before device controller A sees the BG signal, its device wants the bus. What does device controller A do when the BG signal arrives?

Takes the BG signal, does not pass it to B, asserts SACK, and becomes the next bus master.

Note: This Problem Uses the Bus Described In class.

Is this a race condition (yes/no) Explain.

No. A asserted BR before it received BG, so B does not get the BUS this cycle. There is no improper operation that results because of this.

Name: Solution

Problem 2 (20 points)

Suppose IEEE decided to extend the Floating Point standard to include a 12 bit representation, keeping all the characteristics of the IEEE Standard intact. They assign bits to exponent and fraction such that the smallest representable positive number (i.e., smallest subnormal) is 1/1024 and the smallest normalized positive number is 1/16.

Part a: How many bits for exponent: 5

How many bits for fraction: 6

What is the value of n in "excess-n" code for exponents: 5

$\frac{1}{16} \rightarrow 1 \times 2^{-4}$, $1 - n = -4$, $n = 5$

$\frac{1}{1024} \rightarrow 0.000001 \times 2^{-4}$ (6 bits)

Part b: How do you represent "minus infinity": 1 11111 000000

Part c: How do you represent the largest normalized number: 0 11110 111111

What is its value: $1\frac{63}{64} \times 2^{25}$

$1\frac{63}{64} \times 2^{30-5}$

Part d: Consider the value 139/128 times 2^8. If we change how many bits we allocate to exponent and fraction, and we insist that the exponent 0 is representated by 011..1, can we represent this value exactly.
If **yes**, identify the number of fraction bits, the number of exponent bits, and show the representation of this value:

If **no**, explain why not, and represent the number according to the scheme of parts a,b,c above. If rounding is required, round to unbiased nearest.

139/128 = 1.0001011 → 7 fraction bits, leaves 4 Exp. bits
Max Exponent is 1110 = 14; Excess is 7; 14-7=7
Thus, not possible.

0 | 0 1101 | 000110

8+5=13= 01101

Name: [redacted] Solution

Problem 3 (25 points) * Note: The Cache is Empty Beforehand.

A byte-addressable cache of fixed total size and fixed line size is implemented both as a k-way set associative cache and also as a fully associative cache. Assume perfect LRU replacement and line allocate on a write miss policies. For this problem, please show your work below.

If the two implementations are presented with the same sequence of addresses for byte accesses, they generate hits/misses as shown below:

Address	R/W	k-way associative (Miss/Hit)	Fully associative (Miss/Hit)
011011100	W	Miss	Miss
011001100	R	Miss	Miss
111000111	R	Miss	Miss
011011000	R	Hit	Hit
010101100	R	Miss	Miss
001111111	R	Miss	Miss
011001111	R	Miss	Hit
111110010	R	Miss	Miss
110100111	R	Miss	Miss
111000011	R	Hit	Miss

hits because of this.

↳ Byte or Block Bits

Part a: How many cache lines are there in the two cache implementations? 6

Part b: The set associative cache is k-way, where k =

- ☐ 1-Way (Direct Mapped)
- ☐ 2-Way
- ☑ 3-Way
- ☐ 4-Way
- ☐ 5-Way
- ☐ 6-Way
- ☐ 7-Way
- ☐ 8-Way
- ☐ 9-Way

(check the appropriate box).

Part c: For the k-way cache, how many tag bits: 5 and index bits: 1

Part d: Recall in the preamble, that this cache does an allocate on write miss. Complete the hit/miss behavior table if the cache did not allocate on a write miss.

Address	R/W	k-way associative (M/H)	Fully associative (M/H)
011011100	W	M	M
011001100	R	M	M
111000111	R	M	M
011011000	R	M	M
010101100	R	M	M
001111111	R	M	M
011001111	R	M	H
111110010	R	M	M
110100111	R	M	M
111000011	R	H	M

Name: Solution

Problem 4 (25 points)

Consider the following **two level** virtual memory system, similar to the VAX:

Virtual Address Space:	512 Bytes	Physical Memory Size:	128 Bytes
User Space Range:	x000 to x0FF	Page Size:	8 Bytes
System Space Range:	x100 to x1FF	PTE Size:	1 Byte

The machine is byte addressable. Assume the system does not include a TLB. The PTE format is as follows:

Part a: How many bits are allocated for the PFN in the PTE?

$\frac{\text{PM Size}}{\text{Page Size}} = \frac{2^7}{2^3} = 2^4$ **Answer:** 4

We wish to execute the instruction:

ADDM R2, R0, R1

where ADDM takes the contents of the memory location specified by R0, adds to it the value in R1, and stores the result in the memory location specified by R2. That is,

M[R2] <— M[R0] + R1

Before this instruction is executed, the following values are stored as shown:

R0: x0B9
R1: x12
UBR (User Space Page Table Base Register): x140

must be part of system page table

Memory:

x00	x70	x20	x85	x40	x78	x60	x8D
x01	x71	x21	x74	x41	x83	x61	x8A
x02	x8C	x22	x75	x42	x84	x62	x86
x03	x88	x23	x81	x43	x79	x63	x7C
x04	x72	x24	x8C	x44	x89	x64	x7D
x05	x82	x25	x76	x45	x7A	x65	x88
x06	x8F	x26	x77	x46	x7B	x66	x7E
x07	x73	x27	x80	x47	x87	x67	x7F

must be part of user Page table.

Name: Solution

Problem 4 continued

Part b: After the insruction was fetched and decoded, six subsequent memory accesses were required to process this instruction. There were no page faults, interrupts or other uhappy events.

Your job: Complete the table below:

Access #	VA	PA	Data	Description
1	N/A	x42	x84	PTE for page Containing ↓ (SS)
2	x157	x27	x80	PTE for page containing M[R0] (US)
3	x089	x01	x71	Operand Data for ADDM
4	N/A	x42	x84	PTE for page containing ↓ (SS)
5	x154	x24	x8C	PTE for page containing M[R2]/value to get Sum. (US)
6	x0A4	x64	x83	Sum of M[R0]+R1

↑ Sum: x71 + x12 = x83

Note: "N/A" is a potential answer for entries in the VA column. Data is the data that is Read from or Written to memory. Use the Description column to describe what is being read/written (e.g. "PTE for page xx," "Operand data for ADDM").

Part c: What is the value of the SBR (System Space Page Table Base Register)?

Answer: x38

Part d: What is the maximum number of Page Faults ANY execution (excluding instruction fetch) of the ADDM instruction could generate?

x089 = 0|1011|1001 (x17)

x17 + UBR = x17 + x140 = x157

x157 = 1|0101|0111 (x0A)

x0A + SBR = x42, SBR = x42 - x0A = x38

M[x42] = x84 = 1000 0100 (PFN) 111 → x27

M[x27] = x80 = 1000 0000 (PFN) 001 → x01

x64 = b 0110 0100 → Frame xC, look at M[x24]

x24 on same page as x27, see M[x42] 7

Answer: 4

System Page Table is always Resident in Physical Memory.

VA = R2 = x0 PN (100) ← (from x64)

PN + UBR = PN + x140 = b1, PTE_PN, (100) ← (from x2

PTE_PN + SBR = PTE_PN + x38 = x42, PTE_PN = x0A

b1 01010 100 = x154

PN + x140 = x154, PN = x154 - x140 = x14