

NAME _____
First Last

Student ID# _____

STONY BROOK UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
MIDTERM #1 EXAMINATION
VERSION B

CSE 220
Fall Semester 2007
October 11, 2007

This is a closed-book exam. (80 minutes)
Use this form for your work and return it.

The exam has 12 problems.
It is crucial to show all work done on the provided paper.

#1 _____ (12 pts)	#7 _____ (6 pts)
#2 _____ (6 pts)	#8 _____ (6 pts)
#3 _____ (4 pts)	#9 _____ (10 pts)
#4 _____ (6 pts)	#10 _____ (8 pts)
#5 _____ (8 pts)	#11 _____ (8 pts)
#6 _____ (12 pts)	#12 _____ (14 pts)

TOTAL _____ (100 max)

- 1 (i) (3 pts) Convert B1.3E in hexadecimal to base 4. Show all steps.

10110001.00111110
Group by 2's for base 4
10 11 00 01 . 00 11 11 10
Covert to base 4 values (0,1,2,3)
2301.0332

- (ii) (3 pts) Convert B1.3E in hexadecimal to binary. Normalize for single precision IEEE format. (Write answer in the form of $\text{sign} * \text{significand} * 2^{\text{exponent}}$)

10110001.00111110
 $(-1)^0 * 1.0110001001111110 * 2^7$

- (iii) (3 pts) State the sign bit, exponent, and significand in single precision binary form for the normalized number in (ii).

sign bit = 0

exponent = 7, convert to excess 127, $7 + 127 = 134$, 134 in binary = 10000110

significand 01100010011111000000000

IEEE Format: 0 10000110 01100010011111000000000

- (iv) (3 pts) Convert 13.8 in decimal to base 3. Show all steps to get full/partial credit. You must have at least two digits after the decimal point.

$13/3 \rightarrow R 1$

$4/3 \rightarrow R 1$

$1/3 \rightarrow R 1$

Reading upwards $\rightarrow 111$

.8 to base 3 $.8 * 3 = 2.4 \rightarrow 2$

.4 * 3 = 1.2 $\rightarrow 1$

.2 * 3 = 0.6 $\rightarrow 0$

.6 * 3 = 1.8 $\rightarrow 1$

.8 * 3 = repeating

Reading downwards .8 $\rightarrow .\overline{2101}$

111. $\overline{2101}$

2 (6 pts) Consider a decimal number -9. We want to store it in a 7-bit long word that includes the sign bit. What are all seven bits for the following representations of -9?

- (i) Signed Magnitude: **1001001**
- (ii) 1's complement: **0001001, invert 1110110**
- (iii) 2's complement: **add 1 to 1's complement 1110111**

3 (4 pts) What is the largest 2-digit number in base 13? (Answer in base 13). Convert the base 13 number to decimal.

Base 13 can have 13 symbols 0-9, A, B, C.

Therefore the largest number is : CC

$$12 * 13^1 + 12 * 13^0 = 168_{10}$$

4 (6 pts) Which of the following number format(s) have positive and negative representations for zero? Circle each.

- **Sign Magnitude**
- **1's complement**
- 2's complement
- **IEEE single precision floating point**
- **IEEE double precision floating point**

5 Consider two signed numbers A and B. Numbers are six bits long including the sign bit. Negative numbers are stored in their 2's complement form. $A = 100100$, and $B = 111101$.

- (i) (2 pts) What is decimal value of B?
111101, invert 000010, add 1 000011 = -3
- (ii) (2 pts) What is the result of $A - B$ in binary form?
100100 + 000011 = 100111
- (iii) (2 pts) State your answer (for $A - B$) in decimal form.
100111 invert = 011000 = -25
- (iv) (2 pts) Did an overflow occur? If No, for what values of B would overflow occur. If Yes, for what values of B would overflow not occur.
No. Overflow would occur for $B > 4$ or $B < -59$ (not possible in 6 bits)

6 Consider the ASCII character string "MIPS IS FUN". The string is null terminated and stored in memory starting at word address 120 (starts at byte 480). Assume 4 bytes to a word and each word is 32 bits.

(i) (3 pts) Fill in the layout of memory bytes and words that store the above string in a little endian memory. Do not include quotes, but the string-terminating null character is to be included. Show all relevant bytes and words with their addresses and contents (in ASCII).

Word/Byte Address				
120/480	S	P	I	M
121/484		S	I	
122/488	∅	N	U	F
123/492				

(ii) (3 pts) Fill in the layout using little endian memory with the HEX representation of the ASCII characters. Do not include quotes, but the string-terminating null character is to be included. Show all relevant bytes and words with their addresses and contents (in HEX).

Word/Byte Address				
120/480	0x53	0x50	0x49	0x4D
121/484	0x20	0x53	0x49	0x20
122/488	0x00	0x4E	0x55	0x46
123/492				

(iii) (6 pts) Consider the memory in part (ii). Assume that we want to treat each word as a signed integer value instead of ASCII characters.

(a) What is the sign of the value in the second word? 0

(b) What would the sign of the first word be if the values were stored in a big endian memory? 0

(c) What byte is character 'F' stored in? **488**

- 7 (i) (3 pts) When in the Fetch-Execute-Decode cycle of the John von Neumann machine is the Program Counter (PC) incremented? Why?
- (1) Fetch the instruction.
 - (2) Update the PC ($PC = PC + 1$).
 - (3) Decode the instruction.
 - (4) Execute the instruction.

Must occur after Fetch, but before Execute. Once the instruction has been fetched, the instruction is in the IR. Therefore the PC can change without effecting decode. If the instruction is a jump instruction, when the instruction is executed, the PC will be updated with the new address of the next instruction. If the PC is incremented after Fetch the jump address would be incremented by 1 and an instruction would be skipped.

- (ii) (3 pts) Consider the following cycle order:

- (1) Fetch the instruction.
- (2) Decode the instruction.
- (3) Update the PC ($PC = PC + 1$).
- (4) Execute the instruction.

Will this ordering function properly? Why?

Yes it will. See explanation above.

8 (6 pts) Consider a machine that is similar to the von Neumann machine, except that one word contains only one instruction. Also the size or length of each word is the same as the size of an instruction. Each instruction has a 9-bit opcode. Remaining bits in an instruction are for an address. It has 10,240 (10K) words of memory.

(i) What would be the size of MAR?

$$\log(10k) = 14 \text{ bits}$$

(ii) What would be the word size?

$$9 + 14 = 23 \text{ bits}$$

(iii) How many different instructions can this machine have?

$$2^9 = 512 \text{ instructions}$$

9 (i) (5 pts) Cache memory has a main advantage and a main disadvantage over main memory. What are they?

Advantage: Faster, closer to CPU

Disadvantage: More expensive

(ii) (5 pts) What is a split cache? What is the advantage of a split cache over a single, unified cache? **A split cache, is a separate cache for the instructions and for the data. The advantage is that you have 2x the principle of locality. Therefore the split cache will have a higher hit rate.**

- 10 (i) (2 pts) For a register machine, we refer to a register in an instruction using its number. Suppose we have 75 registers in our processor. How many bits do we need to refer to a register?

$$2^7 = 7 \text{ bits}$$

- (ii) (2 pts) Usually, more registers imply faster overall operation. If that is the case, then why do not we have hundreds of registers inside a processor?

registers are more expensive (cost, silicon size). More registers means more bus lines to address them. Additionally, extra registers implies higher amount of logic to manage reading/writing.

- (iii) (4 pts) Floating point registers of MIPS are used for storing single precision numbers and/or double precision numbers. How many floating point registers does MIPS have? If we are using nine registers for nine single precision numbers, how many double precision numbers that we can store in remaining floating point registers?

**32 registers for FP numbers
9 registers leaves 23 registers remaining
need 2 registers per double precision FP
11 double precision FP can be stored**

11 (8 pts) Consider the following C code.

```
#include <stdio.h>

char* mystery1(char *c, int* i) {
while ( *c != NULL && *c != 's') {
*i = *i+1;    c++;}
return c;
}

void mystery2(char* ptr) {
char* temp1 = ptr;    char temp2;

while(*temp1 != NULL)
temp1++;

while(ptr != temp1){
temp1--;
temp2 = *ptr;
*ptr = *temp1;
*temp1 = temp2;
ptr++;
}
return;
}

int main(void){
char str[200];
char* ptr;
int j = 0;

printf ("Enter a string without spaces\n");
scanf ("%s", str);

ptr = mystery1(str, &j);
printf ("%d\n", j);    // Label A

mystery2(ptr);
printf ("%s\n", str);    // Label B
return 0;
}
```

(i) In 2 sentences, what does this program do?

Counts the length of the string to the end or to the letter 's'. Swaps all characters from the 's' on, if the 's' occurs. If no 's', leave the string untouched.

(ii) What does the program print at label A and B for the input string '123strAbC'?

A: 3,

B: 123CbArts

- 12 (14 pts) Consider an array A of ten integers in MIPS memory. Assume that the address of A is in register \$s0. Suppose we want to add the first two elements of the array, subtract the last, and store the value in variable SUM. The address of SUM is in register \$s2. Write the corresponding MIPS instructions to perform the operation. You may use temporary registers. (In other words, write MIPS code for $SUM = A[0] + A[1] - A[9]$;))

```
lw $t0, ($s0)      # load A[0]\\
lw $t1, 4($s0)     # load A[1]\\
lw $t2, 36($s0)    # load A[9]\\

add $t3, $t0, $t1  # $t3 = A[0] + A[1]\\
sub $t3, $t3, $t2  # $t3 = $t3 - A[9]\\

sw $t3, $s2
```

Extra page provided as additional work space.