CS/ECE 252: INTRODUCTION TO COMPUTER ENGINEERING

UNIVERSITY OF WISCONSIN—MADISON

Prof. Gurindar Sohi, Kai Zhao
TAs: Annie Lin, Mohit Verma, Neha Mittal, Daniel Griffin, Yuzhe Ma

Examination 1
In Class (50 minutes)
Monday, September 28, 2016
Weight: 17.5%

NO: BOOK(S), NOTE(S), CALCULATORS OR ELECTRONIC DEVICES OF ANY SORT.
The exam has **nine pages**. You **must turn in the pages 1-8. Circle your final answers**. Plan your time carefully since some problems are longer than others. Use the blank sides of the exam for scratch work.

LAST NAME: ______________________________
FIRST NAME: ______________________________
Section: ______________________________
ID#: ______________________________

Problem	Maximum Points	Points Earned
1	1	
2	3	
3	2	
4	2	
5	4	
6	4	
7	6	
8	4	
9	4	
10	3	
Total	33	

Problem 1 **(1 point)**

Which of the following statements are true?

i) In logic circuits, there is only one way to implement any particular function (e.g. addition).

ii) There can be many different microarchitectures implementing a single type of ISA.

iii) The translation from the unique assembly language of a computer to its ISA is done by an assembler.

Select one answer below:

a) i & ii

b) i & iii

c) ii & iii

d) i, ii & iii

Problem 2 **(3 points)**

Match the following computer abstraction layers with their definitions specified in the table.

a) ISA
b) Program
c) Algorithm
d) Microarchitecture
e) Devices

C	A step by step procedure that is guaranteed to terminate, such that each step is precisely stated and can be carried out by the computer
A	The complete specification of the interface between programs that have been written and the underlying computer hardware that must carry out the work of those programs.
E	Specific hardware technology used to implement logical components of the computer system. (Ex. CMOS/NMOS transistors)
B	A ‘mechanical language’ that is used to specify a sequence of instructions to a computer.
D	A detailed organization of the implementation of an ISA.

Problem 3 **(2 points)**

(a) What is the difference between high level and low level languages?

High level languages are easier to comprehend and are machine independent.

(b) Why are natural languages unsuitable for programming on a computer?

They are ambiguous and thus a computer cannot interpret them.

Problem 4 **(2 points)**

A class has 185 students enrolled in it. Each student is to be assigned a unique ID number (in unsigned binary notation).

(a) What is the minimum number of bits needed to assign each student a unique binary ID number?

2^7 = 128
2^8 = 256 where 128 < 185 < 256
hence min 8 bits

(b) If we were to increase the number of bits by 1, how many more students can be accommodated in the class and still be assigned a unique binary ID number?

Number of students that can be accommodated in 9 bits = 2^9 = 512
Number of students that can be accommodated additionally = 327

Problem 5 **(4 points)**

(a) Using 8 bits for each number, write the 1’s complement, 2’s complement, and signed magnitude binary number of the decimal numbers in the table below:

Decimal	1’s Complement	2’s Complement	Signed Magnitude
19	00010011	00010011	00010011
-10	11110101	11110110	10001010

(b) Why do computers more often use 2's complement notation over 1's complement notation?

It’s easier for arithmetic circuits to calculate.
2’s complement has one representation for zero whereas 1’s complement has only 1

Problem 6 **(4 Points)**

Convert the following 32-bit single-precision IEEE floating point number into decimal value, show your work for full credit:

11000001001001100000000000000000

The bits for the IEEE single-precision floating point number (N) are allocated as follows:

Sign (1 bit)	Exponent (8 bits)	Fraction (23 bits)

Where the value N = $(-1)^{\text{Sign}}$ x 1.Fraction x $2^{\text{Exponent-127}}$

-10.375

Problem 7 **(6 points)**

Perform binary arithmetic for the following numbers as directed. All the numbers are represented in 2's complement form.

(a) Add the following 2's complement numbers. Write the final result as a 6-bit number.

```
  010110
+   1001
--------
  001111
```

Did an overflow error occur in the previous addition? State how you know.

No overflow because overflow cannot occur when adding a positive number with a negative number

(b) Subtract the following 2's complement numbers. Write the final result as a 6-bit number.

```
  010011
- 010010
--------
  000001
```

Did an overflow error occur in the previous operation? State how you know.

No overflow because overflow cannot occur when subtracting a negative number from a negative number

(c) Convert the answers for (a) and (b) into decimal numbers. Show your work for full credit.

001111=8+4+2+1=15
000001=1

Problem 8 **(4 points)**

(a) Convert the following 3 hexadecimal numbers to unsigned binary. A 0x before a set of symbols (from 0 to F) indicates a hexadecimal number.

0x24 = 0010 0100

0xAA = 1010 1010

0xD7 = 1101 0111

(b) Perform the specified logical operations on the following hexadecimal numbers. Write your result in unsigned binary. (Hint: first convert the hexadecimal to binary, which you did in part (a), and then perform the operations)

NOT(0x24 OR (0xAA AND 0xD7))

0101 1001

(c) Convert the result of part (b) into hexadecimal.

0x59

Problem 9 **(4 points)**

The ASCII table on the last page will be useful in solving this problem. You can detach it to make it easier to consult without flipping pages.

Consider two strings of two ASCII characters each: "**64**" and "**%@**". The ASCII characters in the string are the two characters between the quotation marks, and there is no null character terminating the string.

(a) First, convert each of these two strings into their corresponding 16-bit binary values.

64 = 3634 = 0011 0110 0011 0100

%@ = 2540 = 0010 0101 0100 0000

(b) Now, suppose that the two 16-bit binary values were added as if they were signed integers in 2's complement, and the resulting 16 bits treated as they were a string of ASCII characters. What would the resulting ASCII string be? Show your work for full credit.

Sum = 0101 1011 0111 0100 =5B74

ASCII string = [t

Problem 10 **(3 points)**

Give an example of an *integer* that can be represented in floating point format (32-bit IEEE format), but cannot be represented as a 32-bit 2's complement integer. Show its 32-bit IEEE format representation.

The biggest 2's complement integer that can be represented with 32 bits is 2^31 -1, so any integer number greater than 2^31 -1, but less than 2^128 , is the answer to this question.

so 2^31 , 2^31 +1, 2^31 +2 and 2^32 are some examples. Then convert it to IEEE.

ASCII Table

Character	Hex	Character	Hex	Character	Hex	Character	Hex
nul	00	sp	20	@	40	`	60
soh	01	!	21	A	41	a	61
stx	02	“	22	B	42	b	62
etx	03	#	23	C	43	c	63
eot	04	$	24	D	44	d	64
enq	05	%	25	E	45	e	65
ack	06	&	26	F	46	f	66
bel	07	‘ *(Apostr.)*	27	G	47	g	67
bs	08	(	28	H	48	h	68
ht	09	)	29	I	49	i	69
lf	0A	*	2A	J	4A	j	6A
vt	0B	+	2B	K	4B	k	6B
ff	0C	, *(Comma)*	2C	L	4C	l	6C
cr	0D	-	2D	M	4D	m	6D
so	0E	. *(Period)*	2E	N	4E	n	6E
si	0F	/	2F	O	4F	o	6F
dle	10	0	30	P	50	p	70
dc1	11	1	31	Q	51	q	71
dc2	12	2	32	R	52	r	72
dc3	13	3	33	S	53	s	73
dc4	14	4	34	T	54	t	74
nak	15	5	35	U	55	u	75
syn	16	6	36	V	56	v	76
etb	17	7	37	W	57	w	77
can	18	8	38	X	58	x	78
em	19	9	39	Y	59	y	79
sub	1A	:	3A	Z	5A	z	7A
esc	1B	;	3B	[	5B	{	7B
fs	1C	<	3C	\	5C	|	7C
gs	1D	=	3D	]	5D	}	7D
rs	1E	>	3E	^	5E	~	7E
us	1F	?	3F	_ *(Undrscre)*	5F	del	7F