CS/ECE 252: INTRODUCTION TO COMPUTER ENGINEERING

Spring 2015, Section 2

UNIVERSITY OF WISCONSIN—MADISON

Prof. Mark Hill

TAs: Sujith Surendran, Lisa Ossian

Midterm Examination 1

In Class (50 minutes)

Friday, February 13, 2015

Weight: 17.5%

NO: BOOK(S), NOTE(S), OR CALCULATORS OF ANY SORT.

The exam has **nine** pages. **Circle your final answers**. Plan your time carefully since some problems are longer than others. You **must turn in the pages 1-8**. Use the blank sides of the exam for scratch work.

Note: ASCII table is provided on Page 9

LAST NAME: ____

FIRST NAME: ____

ID#: ____

Problem	Maximum Points	Points Earned
1	1	
2	1	
3	1	
4	3	
5	2	
6	2	
7	3	
8	2	
9	4	
10	4	
11	2	
12	4	
13	1	
Total	30	

Problem 1 **(1 Point)**

How does a microarchitecture differ from an instruction set architecture (ISA)?

The microarchitecture specifies how circuits are put together to create the computer. The Instruction Set Architecture (ISA) provides an interface which specifies what sort of instructions a computer supporting this interface can perform. We would do this for a number of reasons. Primarily, requirements could be different for different systems.

Problem 2 **(1 Point)**

Explain briefly why natural languages cannot be used as programming languages.

Natural Languages are ambiguous.

Problem 3 **(1 Point)**

Mention one difference between high-level languages and assembly languages.

High level languages are “machine independent”, whereas the assembly languages are dependant on the machine on which it is executed.

Problem 4 **(3 Points)**

Label the following items/terms according to their level of abstraction relative to one another. Label the most abstract term as 1 and least abstract as 6.

4	Instruction set architecture (ISA)
3	Code in high-level language (C/C++/Java)
1	Problem statement/application
6	Transistors (CMOS/NMOS/PMOS)
2	Algorithm to solve the problem
5	Microarchitecture

Problem 5 **(2 Points)**

Shown below are a few concepts covered in Chapter 1:

- Definiteness
- Finiteness
- Effective Computability
- Abstraction
- Language/Code

In the table below, fill in the name of the concept that best matches the corresponding description:

Concept	**Description**
Effective Computability	Determines whether or not a problem is solvable
Abstraction	Underlying mechanisms are hidden or unknown
Finiteness	Will not run on forever, will stop at some point
Definiteness	Each step of a process must be clearly laid out
Language/Code	Can be used to write an algorithm that a computer can understand

Problem 6 **(2 Points)**

When CS252 was offered in Spring 1999, assume that only 6 bits were required to uniquely represent every student enrolled in this course. However, assume that the number of students who will be enrolling this course in Spring 2016 will be 5 times the number of students enrolled in Spring 1999. What is the minimum number of bits required to uniquely represent every student who will be enrolled in Spring 2016?

Number of students enrolled in Spring 1999 = 2^6 = 64

Number of students who will be enrolling in Spring 2016 = 64*5 =
Number of bits required to represent 320 students = log2(64*5) = 9

Problem 7 **(3 Points)**

Using 8 bits to represent each number, write the representations of -4, 97 and -97 in signed magnitude, 1's complement, and 2's complement notations.

Number	Signed Magnitude	2's complement
-4	1000 0100	1111 1100
97	0110 0001	0110 0001
-97	1110 0001	1001 1111

Problem 8 **(2 Points)**

Fill in the table below with the largest and smallest decimal numbers that can be represented with:

a) 13-bit unsigned number
b) 13-bit 2's complement number
(Note: -2 is smaller than -1)

Representation	Largest decimal number that can be represented	Smallest decimal number that can be represented
13-bit unsigned number	**2^12 - 1 = 4095**	**-(2^12 -1) = -4095**
13-bit 2's complement number	**2^12 - 1 = 4095**	**-(2^12) = -4096**

Problem 9 **(4 Points)**

Perform the specified logical operations on the following 16-bit numbers expressed in hexadecimal representation. Express your result in **hexadecimal** (base 16).

a. xA007 OR xBF09

Answer: xBF0F

b. NOT(xA005) AND xFFF4

Answer: x5FF0

Problem 10 **(4 Points)**

Perform binary arithmetic for the following pairs of 2's complement numbers. Write your result in binary. Also indicate if there is any overflow.

a.
```
  1100 1000
+ 1111 0111
  1011 1111
```

Is there any overflow? Yes.

b.
```
  1011 0000
- 0001 0001
  1001 1111
```

Is there any overflow? No.

Problem 11 **(2 Points)**

Assume that we have an 8-bit fixed point binary notation, with 5 bits for the integer part, i.e., 5 bits to the left of the binary point, and 3 bits for the fractional part, i.e., 3 bits to the right of the binary point. Represent the decimal 14.5 in this fixed point notation.

14 = 1110
.5 = 5/10 = ½ = 0.1
=> 14.5 = 01110.100

Problem 12 **(4 Points)**

Convert the decimal value -23.125 into its IEEE single-precision floating point representation.

23 = 10111

.125 = ⅛ = 0.001

=> -23.125 = - (10111.001) = -(1.0111001*2^4)

=> Exponent = 127+4 = 131, Fraction = 0111001, Sign = 1

=> Answer = 1 10000011 01110010000000000000000

Problem 13 **(1 Point)**

Convert the ASCII string "Spr_15" to its hexadecimal representation. Only represent the characters within the quotation marks and assume it is null-terminated.
(Hint: See ASCII-to-hexadecimal table on the last page of the exam.)

Answer: 53 70 72 5F 31 35 00

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