CPSC 121: Models of Computation
SAMPLE Midterm, 2009 February 24

Name: ______________________ Student ID: ______________________

Signature: ______________________

- The cover page on the real midterm will be identical to this one except that (1) it will not include this line, (2) the number of questions/marks below will be specified, and (3) it will have an area on the page for us to indicate your mark on each question and overall. **Read these instructions now!**

- You have **90 minutes** to write the ?? questions on this quiz.

- A total of **?? marks** are available. You may want to complete what you consider to be the easiest questions first!

- Ensure that you clearly indicate a single legible answer for each question.

- You are allowed any reasonable number of textbooks and a single binder or folder of notes as references. Otherwise, no notes, aides, or electronic equipment are allowed.

- Good luck!

UNIVERSITY REGULATIONS

1. Each candidate must be prepared to produce, upon request, a UBCcard for identification.

2. Candidates are not permitted to ask questions of the invigilators, except in cases of supposed errors or ambiguities in examination questions.

3. No candidate shall be permitted to enter the examination room after the expiration of one-half hour from the scheduled starting time, or to leave during the first half hour of the examination.

4. Candidates suspected of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action:

   - having at the place of writing any books, papers or memoranda, calculators, computers, sound or image players/recorders/transmitters (including telephones), or other memory aid devices, other than those authorized by the examiners;
   - speaking or communicating with other candidates; and
   - purposely exposing written papers to the view of other candidates or imaging devices. The plea of accident or forgetfulness shall not be received.

5. Candidates must not destroy or mutilate any examination material; must hand in all examination papers; and must not take any examination material from the examination room without permission of the invigilator.

6. Candidates must follow any additional examination rules or directions communicated by the instructor or invigilator.

Notes about this Sample Exam

Along with the *many* other practice resources available to you in the textbook and on the course website, this sample midterm is intended to prepare you for the upcoming midterm on February 24, 2009. We *strongly* recommend working through this sample midterm (although if that's your only preparation for the exam, you will likely find that it's insufficient).

Although the real midterm will differ from this sample, its structure and some of the features of its questions will be similar. We note particular similarities in footnotes on each question.

1 Representing Relationships with Predicate Logic[1]

Let S be the set of all students and C be the set of all courses. Consider the following predicates:

- $Required(s, c)$ means course c is required for student s to graduate
- $Enrolled(s, c)$ means student s is enrolled in course c
- $Full(c)$ means course c has no room for further enrollment

1. Translate the following into predicate logic:

(a) If a course is full, then everyone for whom the course is required is enrolled in that course.

(b) There are no full courses.

2. Translate the following into English:

(a) $\sim\exists s \in S, \exists c \in C, Enrolled(s, c) \wedge Required(s, c)$.

(b) $\forall s \in S, \exists c_1 \in C, \exists c_2 \in C, c_1 \neq c_2 \wedge Enrolled(s, c_1) \wedge Enrolled(s, c_2)$.

3. Negate the statements from part 2. Write two versions: the negation changing as little of the form of the original statement as possible, and the negation after using the double negation and De Morgan's laws to move the negations "inward" as far as possible.

[1] If the midterm contains a similar question, it will use exactly the same definitions (S, C, $Required$, $Enrolled$, and $Full$).

4. Give a particular set of students and courses and truth values for $Enrolled$, $Full$, and $Required$ such that the second statment from part 1 and the first statement from part 2 are both true.

2 Proof Critique[2]

Indicate the step or steps in the proof below that are invalid or state here that the proof is valid.

(Note: this problem uses a syntax for negation in which $\sim p$ is represented as $\overline{p}$.)

Let $\mathbf{Z}^+$ be the set of all positive integers.

Proof (by contrapositive): $\forall n \in \mathbf{Z}^+$, if n^2 is even then n is even.

Recall: $p \rightarrow q \equiv \overline{q} \rightarrow \overline{p}$

Thus, in order to prove

$$\forall x, P(x) \rightarrow Q(x)$$

we can prove

$$\exists x, \overline{Q(x)} \rightarrow \overline{P(x)}$$

Let $E(x)$: x is even

To prove $\forall n \in \mathbf{Z}^+, E(n^2) \rightarrow E(n)$ we prove $\exists n \in \mathbf{Z}^+, \overline{E(n)} \rightarrow \overline{E(n^2)}$ via a direct proof,

Let $n \in \mathbf{Z}^+$ be an odd integer.

Therefore, by definition of odd, $n = 2k+1$ for some $k \in \mathbf{Z}^+$.

Now,

$$\begin{aligned} n^2 &= (2k+1)^2 \\ &= 4k^2 + 4k + 1 \\ &= 2(2k^2 + 2k) + 1 \end{aligned}$$

Since the sum and product of integers is an integer, we see that $m = 2(2k^2 + 2k)$ is an even integer and $n^2 = m + 1$ is an odd integer QED

[2]The midterm may contain a problem that also asks about critiquing a proof.

3 Number Representation[3]

Since you do not have a calculator, you may leave your answers as arithmetic expressions (e.g., 53 + 27 + 5), but the correct answers should be obtainable from your expressions simply by entering them into a basic calculator.

1. What's the minimum number of bits required to represent the decimal number -23 as a signed binary number? Convert -23 to a signed binary number with that many bits.

2. What's the minimum number of bits required to represent the decimal number 23 as an unsigned binary number? Convert 23 to an unsigned binary number with that many bits.

3. Explain the relationship between your answers for the previous two parts (i.e., both how the number of bits compare and how the bit patterns compare).

4. Convert the unsigned binary number 1101001 to (a) octal and (b) hexadecimal.

5. If we use 16 bits to represent non-negative numbers, dividing the 16 bits into 12 bits to the left of the "decimal place" and 4 to the right, what's the smallest value larger than 0 that we can represent?

6. Why is it inaccurate to say that hexadecimal is a "more compact" way to store a number in a typical modern computer than binary?

[3] The midterm may contain problems that ask about number representation. If it does, the same allowance will be made for arithmetic given the lack of calculators.

4 Circuit Properties[4]

For each of the following statements, label them as **definitely true**, **possibly true**, or **definitely false** for any combinational circuit that can be modeled by one or more propositional logic statements. (That is, circle one of those options for each statement.)

Note: combinational circuits are the circuits we worked with in the first part of the course in which no path through the circuit loops back on itself.

1. Every input of every gate is connected to either the output of a gate, an input to the circuit, or "power" or "ground". (Recall that "power" is HIGH, 1, or true while "ground" is LOW, 0, or false.)

Circle one: **definitely true** **possibly true** **definitely false**

2. Some gate's output is connected to an input of another gate.

Circle one: **definitely true** **possibly true** **definitely false**

3. No two gates' inputs are connected to each other.

Circle one: **definitely true** **possibly true** **definitely false**

4. No input to the circuit is connected to the output of a gate.

Circle one: **definitely true** **possibly true** **definitely false**

[4] If the midterm contains a question about circuit properties, it will use the same descriptive text and format (but different statements to classify).

5 Circuit Design[5]

It's unknown whether the following well-studied sequence either terminates or enters a repeating loop for every starting point (i.e., every positive integer n):

If n is 1, stop.
Otherwise, if n is divisible by two, divide n by two and repeat.
Otherwise, multiply n by three, add one, and repeat.

For example, if $n = 17$ initially:

Step 0:	$n = 17$	for next step, use $n * 3 + 1$
Step 1:	$n = 52$	for next step, use $n/2$
Step 2:	$n = 26$	for next step, use $n/2$
Step 3:	$n = 13$	for next step, use $n * 3 + 1$
Step 4:	$n = 40$	for next step, use $n/2$
Step 5:	$n = 20$	for next step, use $n/2$
Step 6:	$n = 10$	for next step, use $n/2$
Step 7:	$n = 5$	for next step, use $n * 3 + 1$
Step 8:	$n = 16$	for next step, use $n/2$
Step 9:	$n = 8$	for next step, use $n/2$
Step 10:	$n = 4$	for next step, use $n/2$
Step 11:	$n = 2$	for next step, use $n/2$
Step 12:	$n = 1$	

At this point, the sequence terminates.

Using the steps below, design a circuit that inputs an 8-bit unsigned binary number and outputs the next number in this sequence.

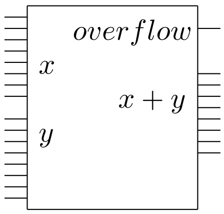
You may design your circuit using inverters, 2-input XOR and XNOR gates, 2- or 3-input AND, OR, NAND, and NOR gates, 2:1 multiplexers, and the modules described below.

1. Design a circuit that takes an 8-bit unsigned binary number as input and divides it by 2 (rounding down if necessary). Your circuit should have one 8-bit output, the result.

[5]If the midterm contains a circuit design problem: (1) it will have a shorter introductory "story", (2) it won't require drawing quite so many wires, (3) it will likely also involve a multi-step question in which some previous step or steps is used as a module in a later step, and (4) it will allow the same set of gates, although any extra modules allowed (like the 8-bit adder) may be different.

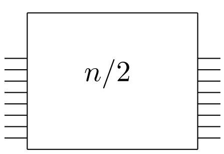
2. Design a circuit that takes an 8-bit unsigned binary number as input, multiplies it by 3, and adds 1. In addition to the circuit elements described above, you may use 8-bit adders.

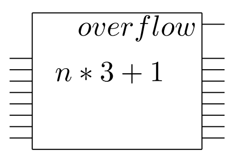
An 8-bit adder takes two 8-bit unsigned binary numbers as input and has two outputs: the 8-bit binary sum and a Boolean value indicating whether overflow occurred. Use this symbol for an 8-bit adder:



Your circuit should output the 8-bit result and a Boolean value indicating whether or not the output is correct (given the possibility of overflow). Hint: consider that $x * 3 = x * 2 + x$.

In the next part, use your circuit from parts 1 and 2 as modules with the following symbols:





You should use these modules in part 3, even if you do not complete the previous parts. Assume the modules work as described in the problem statement for the previous parts.

3. Design a circuit that inputs an 8-bit unsigned binary number and outputs the next number in this sequence if possible. Its outputs are the 8-bit result and a Boolean value indicating whether or not the output is correct (given the possibility of overflow).

4. Speculate as to whether 8 bit numbers will be large enough to find a number on which the sequence neither terminates nor enters a repeating loop.

6 Proving Properties of Computing Systems[6]

A decision algorithm is an algorithm that, when run on an input, answers either yes or no. Furthermore, there is a "correct" answer for each input, either yes or no, and the algorithm may or may not get the correct answer.

A decision algorithm is called sound exactly when: if the algorithm reports that the answer for an input is yes, the correct answer for that input is yes.

A decision algorithm is called complete exactly when: if the correct answer for an input is yes, then the algorithm reports that the answer for that input is yes.

(An algorithm that is not sound is "unsound". An algorithm that is not complete is "incomplete".)

Consider the following procedure to generate a new decision algorithm, which we call "inverting": the inverse algorithm to a_1 is $\widehat{a_1}$. $\widehat{a_1}$ answers yes if and only if a_1 answers no.

1. Prove that if there's at least one input that an algorithm operates on (i.e., answers either yes or no to) whose correct answer is no, then the algorithm and its inverse cannot both be sound.

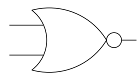
2. Prove that if there's at least one input that an algorithm operates on whose correct answer is yes, then the algorithm and its inverse cannot both be complete.

3. Prove that an algorithm and its inverse may both be unsound.

[6] If the midterm contains a similar problem, it will use exactly the same definitions of "soundness" and "completeness" but different theorems to prove and likely a different procedure for generating an algorithm.

7 Universality[7]

The task is to prove that a 2-input NOR gate is universal. The gate symbol is:



1. What does it means for a logic gate or a collection of logic gates to be universal?

2. Show how a 2-input NOR gate can be used to implement an inverter.

3. Show how a 2-input NOR gate can be used to implement a 2-input AND gate.

4. Show how a 2-input NOR gate can be used to implement a 2-input OR gate.

[7] The midterm may contain a similar question.

8 Abstract Proof[8]

Describe in as much detail as possible what strategy you would use to prove the following theorem using a direct proof approach.

$\forall x \in D, \forall y \in E, P(x, y) \vee x = y.$

[8] If the midterm contains an abstract proof problem, it will be exactly the same in form but with a different statement to prove.