CPSC 121: Models of Computation
SAMPLE Final Exam, 2009 April 17

Name: _____ Student ID: _____

Signature: _____ Section (202, 203, or BCS): _____

• The cover page on the real exam 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 **150 minutes** to write the ?? questions on this exam.

• 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.

This page intentionally left blank.
If you put solutions here or anywhere other than the blank provided for each solution, you must *clearly* indicate which problem the solution goes with and also indicate where the solution is at the designated area for that problem's solution.

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 final is intended to prepare you for the upcoming final exam on April 17, 2009. We *strongly* recommend working through this sample final (although if that's your only preparation for the exam, you will likely find that it's insufficient).

Although the real final 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.

Besides the notes included in footnotes below, please note the following: You can also expect to see a small number of questions closely related to questions that were on the midterm exam. Reviewing the midterm, sample midterm, and midterm followup questions would be wise!

Finally, note that because this is a sample, we are free to ask questions that would be awkward or inappropriate on the final exam. As examples, the DFA design problem on recognizing programming language strings uses characters that are hard to write clearly (quotation marks and backslashes) and that are awkward to include in regular expressions, and the final DFA in that part has far more states than any reasonable DFA to design on the final. With the additional time and resources available to you while working a sample exam, however, this is an excellent practice problem.

1 Functions: 1-1, Onto, and 1-1 Correspondence [SOME marks][1]

For each of the following, indicate whether it is 1-1, onto, and/or a 1-1 correspondence. In some cases, we also ask you to briefly justify your answer.[2]

1. $f : \mathbf{R} \to \{-1, 0, 1\}$, defined by:

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is positive} \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x \text{ is negative} \end{cases}$$

1-1 (Circle one): YES NO

Briefly justify:

Onto (Circle one): YES NO

Briefly justify:

1-1 correspondence (Circle one): YES NO

No need to justify.

2. $f : \{-1, 0, 1\} \to \mathbf{R}$, defined by:

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is positive} \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x \text{ is negative} \end{cases}$$

1-1 (Circle one): YES NO

Briefly justify:

Onto (Circle one): YES NO

No need to justify.

1-1 correspondence (Circle one): YES NO

No need to justify.

---

[1] The exam may have a question with the same structure but different functions.

[2] What is a good brief justification? It's the heart of the proof you *would* write, without all the trappings of the proof. So, to justify that a function is **not** 1-1 or onto, give an explicit counterexample and briefly explain why it's a counterexample. To justify that a function **is** 1-1, show why given two equal images—$f(x) = f(y)$—their pre-images must be equal—$x = y$. To justify that a function **is** onto, for an arbitrary image y, give a pre-image x (which involves multiple variables, in some cases) such that $f(x) = y$.

3. $f(x, y, z) = x + \frac{y}{z}$, where $f : \mathbf{Z} \times \mathbf{Z}^+ \times \mathbf{Z}^+ \rightarrow \mathbf{R}$. *Hint: when answering about "onto", consider* π.

1-1 (Circle one): YES NO

Briefly justify:

Onto (Circle one): YES NO

Briefly justify:

1-1 correspondence (Circle one): YES NO

Briefly justify:

4. $f : P(\mathbf{Z}^+) - \{\{\}\} \rightarrow \mathbf{Z}^+$, defined by $f(S) = min(S)$, i.e., $f(S)$ returns the smallest integer in S. (We assume that it is possible to find the smallest integer in any non-empty subset of the positive integers.)

1-1 (Circle one): YES NO

No need to justify.

Onto (Circle one): YES NO

Briefly justify:

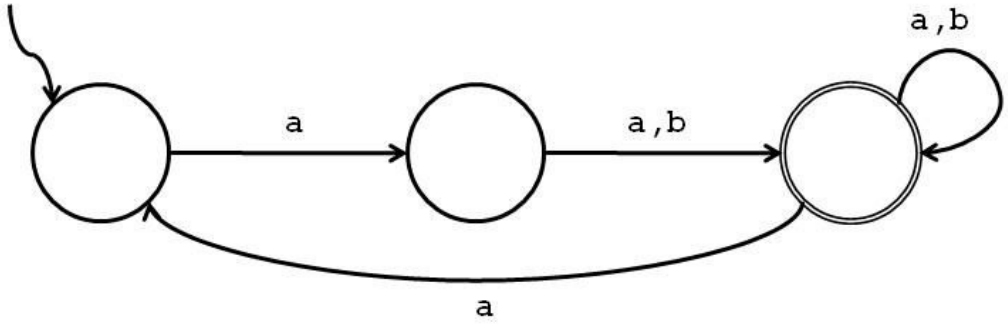
1-1 correspondence (Circle one): YES NO

No need to justify.

2 Broken DFAs [SOME marks][3]

There are two problems with each of the following "DFAs" that keep them from being correct DFAs. For each one, briefly explain each problem in the blank supplied.

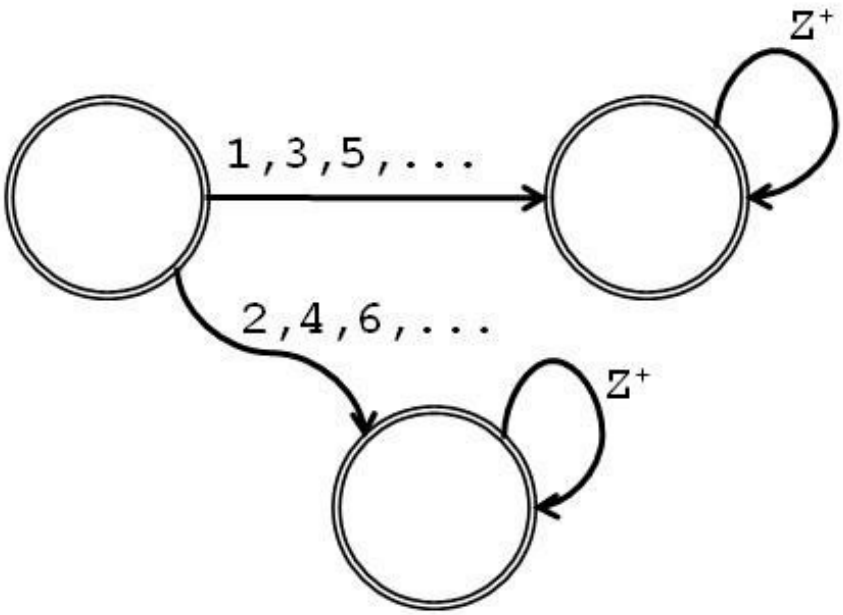
1. Explain what's wrong with the following "DFA" that operates over the alphabet $\{a, b\}$:



Problem #1: _____

Problem #2: _____

2. Explain what's wrong with the following "DFA" that operates over the alphabet $\mathbf{Z}^+$:



Problem #1: _____

Problem #2: _____

[3]The exam may have a question that also explores the connection between the graphical and formal representations of DFAs.

3 Planning Induction Problems [SOME marks][4]

1. Describe how the bounds on n—the induction variable representing the step to be proven in the inductive step—and k—the induction variable representing the steps assumed in the induction hypothesis—generally relate to the set of base cases for a strong induction problem. Assume b_{min} is the smallest integer at which we establish a base case; b_{max} is the largest integer at which we establish a base case; and we also have base cases for every integer between b_{min} and b_{max}.

2. Both of the following strong induction proofs that one can make any amount of change greater than or equal to 18 cents with 4 and 7 cent pieces are both valid. Explain why they are actually equivalent, accounting clearly for each point where they differ. Note: there is no difference in the base cases.

Base cases:
We can make 18 cents as two 7s and a 4.
We can make 19 cents as three 4s and a 7.
We can make 20 cents as five 4s.
We can make 21 cents as three 7s.

Induction Hypothesis:
Assume for integers $n \geq 21$ and integers k where $18 \leq k \leq n$ that we can make k cents of change using only 4 and 7 cent pieces.

Inductive Step:
We must show that we can make $n+1$ cents of change using only 4 and 7 cent pieces.

Note: $n \geq 21$, $n+1-4 = n-3 \geq 18$. So, we can make $n+1-4$ cents of change based on our induction hypothesis.

By adding 4 cents to that amount of change, we can make $n+1$ cents of change. This completes our inductive proof.

Base cases:
We can make 18 cents as two 7s and a 4.
We can make 19 cents as three 4s and a 7.
We can make 20 cents as five 4s.
We can make 21 cents as three 7s.

Induction Hypothesis:
Assume for integers $n > 21$ and integers k where $18 \leq k < n$ that we can make k cents of change using only 4 and 7 cent pieces.

Inductive Step:
We must show that we can make n cents of change using only 4 and 7 cent pieces.

Note: $n > 21$ and so $n \geq 22$, $n-4 \geq 18$. So, we can make $n-4$ cents of change based on our induction hypothesis.

By adding 4 cents to that amount of change, we can make n cents of change. This completes our inductive proof.

[4]The exam may also have a question that explores the process of planning out an induction proof (or other proof type).

4 Algorithmic Induction Proof [SOME marks][5]

We wish to compute the power function, $f : \mathbf{R}^+ \times \mathbf{Z}^+ \to \mathbf{R}^+$, defined by $f(x, n) = x^n$. Consider the following algorithm, `Power(x,n)`, to compute $f(x, n) = x^n$.

`Power(x,n)`

- If `n` $= 1$ then return `x`.
- Otherwise, return `x` times the result of calling `Power(x,n-1)`

1. Does the algorithm `Power(x,n)` terminate (i.e., run to completion) for arbitrary `x` $\in \mathbf{R}^+$ and `n` $\in \mathbf{Z}^+$? Briefly justify your answer.

2. Prove by induction[6] that the algorithm, `Power(x,n)`, is correct. That is, prove

$\forall x \in \mathbf{R}^+, \forall n \in \mathbf{Z}^+,$ `Power`$(x, n) = x^n$

Hint: You will want to choose one variable, x or n, as the induction variable. As you make your choice, consider that x is a real number while n is an integer.

[5] The exam may also have an induction proof question about properties of an algorithm.

[6] In an actual exam, marks are given for writing the framework of the proof clearly and completely. In other words, you are marked on the form of your proof as well as on the content.

5 Induction Proof [SOME marks][7]

Consider the following algorithm for randomizing the order of the elements of a list (which you might use, for example, to shuffle a deck of cards in a computer program). We assume that(1) as with Java arrays, the first element's index is 0 and the last element's index is the length of the list minus one and (2) each outcome of a single random choice is equally likely. (If it helps, the latter assumption means that "pick a random index" is equivalent to "pick an index uniformly at random". If that's unfamiliar, don't worry about it!)

`Reorder(list)`

- If the list is of length 1, leave it as is.
- Otherwise, the list has at least two elements.
- Pick a random index r between 0 and the length of the list minus 1 (inclusive). Swap the element at index 0 with the element at index r. (If $r = 0$, just leave the 0^{th} element where it is.)
- Now, call `Reorder` on everything but the first element of the list (i.e., the list formed by elements with the indexes $1, 2, 3, \ldots,$ length $- 1$).

1. Prove by induction that using this algorithm, every element can end up at any location in the list (for lists of any length one or greater).

[7] The exam may also have a weak induction proof.

2. Prove that using this algorithm, every element has an equal chance to end up at any location in the list (for lists of any length one or greater).[8]

[8] Some tips for working with probability: For every element to have an equal chance of ending up at any location in a list of length n, each element must have a $\frac{1}{n}$ chance of ending up at each location. When we "pick a random number between 0 and $n-1$", we are selecting an integer uniformly at random from the set of integers $\{0, 1, 2, \ldots, n-1\}$. There are n such integers; so, each one has a $\frac{1}{n}$ chance of being chosen and a $\frac{n-1}{n}$ chance of *not* being chosen. The chances that you will first make one random choice with probability p (e.g., $\frac{n}{n+1}$) and then make a second independent random choice with probability q (e.g., $\frac{1}{n}$) are $p \times q$ (or $\frac{n}{n+1} \times \frac{1}{n}$ for our running example). None of this is required CPSC 121 knowledge, although it is necessary to solve this particular problem!

6 Strong Induction [SOME marks][9]

Prove the following theorem using strong induction: Every positive integer greater than or equal to 2 is a prime number, a power of a prime number, or a product of powers of primes.[10]

[9] The exam may have a strong induction question.

[10] Note that $1 = 2^0$; so, we could actually prove this for any positive integer as well.

7 Sets and Functions Proof [SOME marks][11]

1. Prove that if $f : A \to B$ is an injection and $|A|$ and $|B|$ are both finite, then there is no $g : A \to B$ such that g is a surjection but not an injection. *Hint: try contradiction.*

2. If we remove the constraint that $|A|$ and $|B|$ be finite, it becomes possible for $f : A \to B$ to be an injection while $g : A \to B$ is a surjection but not an injection. Prove this by describing witness functions f and g, where $A = B = \mathbf{Z^0}$ and $\mathbf{Z^0}$ is the non-negative integers.

[11] The exam may have a question asking you to prove properties of sets and/or functions, although such a proof will *not* focus on finite and infinite cardinalities as this one does.

8 DFAs and Regular Expressions [SOME marks][12]

1. Design a DFA to recognize Java strings. Your DFA should insist on a `"` to start the string and a `"` to end it and should disallow `"` in the middle. For example, your DFA should accept `""`, `"hello world!"`, and `"'"` but reject `''`, `'"'`, `"""hello "world"!"""`, `"hello "world`, `"hello \\"world\"!"`, `"hello \\\"world\"!"`, and `"hello \"world\"!"`. In this part and the rest of this problem, make any reasonable assumption you need to about the input alphabet.

2. Design a regular expression for the same language.

3. Design a new DFA to recognize strings. Your DFA should insist on `"` to start the string and `"` to end it, but it should allow a `"` in the middle of the string if an odd number of `\` characters comes just before the `"`. (That is, allow "escaped" quotes. The "odd number" restriction is because two consecutive backslashes inside a string are an escape sequence for a single backslash. With an odd number, the leftover one escapes the quote.) For example, your DFA should accept `""`, `"hello world!"`, `"'"`, `"hello \\\"world\"!"`, and `"hello \"world\"!"` but reject `''`, `'"'`, `"""hello "world"!"""`, `"hello "world`, and `"hello \\"world\"!"`.

4. Design a regular expression for the same language. Note that either `\\` or `[\]` matches a single backslash.

[12] The exam may have a question with a similar format focused on designing DFAs and regular expressions.

5. In the Python programming language, strings can be in one of four forms: (1) starting and ending with a single `"`, in which case `"` can only appear in the string if preceded by an odd number of \ characters, (2) starting and ending with a single `'`, in which case `'` can only appear in the string if preceded by an odd number of \ characters, (3) starting and ending with three `"`, in which case `"` can appear in the middle of the string, but escaping is required to put three or more `"` characters in a row in the middle of the string, and (4) starting and ending with three `'`, which works analogously to case (3). Furthermore, the string can be optionally preceded by one or both of two modifier characters: an uppercase or lowercase u and an uppercase or lowercase r, in that order if both appear.

(a) Design a set of test cases for your DFA that should be accepted and rejected. (The ones from the previous problem are a good start.)

(b) Design a DFA to recognize Python strings.

(c) Design a regular expression for the same language.

(We omitted one property of Python strings: a "newline" cannot appear in the single `"` and `'` versions unless escaped by an odd number of \ characters. In the triple-quote versions, newline may appear as a normal part of the string. Feel free to try that as well!)

9 Proof Puzzle [SOME marks][13]

Consider the following "scrambled" proof with a total of 12 lines. When unscrambled, the proof establishes that for all sets A, B, and C, for all functions $f : B \to C$ and $g : A \to B$, if f and g are injective, then $h(x) = f(g(x))$, where $h : A \to C$, is an injective function. Put the 12 lines in the correct order by writing "1" next to the first line of the proof, "2" next to the second line, etc. There may be multiple valid orderings; choose the valid ordering that makes the proof clearest.

_____ Without loss of generality, let A, B, and C be sets.

_____ Since g is a function, there is exactly one image of x under g in B.

_____ Thus, h is injective.

_____ Without loss of generality, let $f : B \to C$ and $g : A \to B$ be functions.

_____ Consider an arbitrary element $x \in A$.

_____ Since g is injective, $x = y$.

_____ Since f is injective, $g(x) = g(y)$.

_____ QED.

_____ Since f is a function, there is exactly one image of $g(x)$ under f in C.

_____ Assume $h(x) = h(y)$.

_____ Then, $f(g(x)) = f(g(y))$.

_____ So, $h : A \to C$ is a function.

[13] The exam may have a question with a similar "proof puzzle" format.

10 Sets and Functions by Example [SOME marks][14]

Consider the following sets $A = \{\emptyset, \text{cs}, 5, 2\}$, $B = \{\text{cs}, A, \{A\}\}$, and $C = \{\}$.

1. Give the values of $|A|$, $|B|$, and $|C|$.

2. For each of the three pairs of sets, list the elements of and the cardinality of each of the following:

   - set1 $\cap$ set2

   - set1 $\cup$ set2

3. Explain why there are more than "three pairs" of ways to plug A, B, and C into set1 $-$ set2. What are the cardinalities of $A - B$ and $A - C$.

4. Which of the following are true? $A \subseteq B$, $B \subseteq C$, $C \subseteq A$, $A \in B$, $B \in C$, and $C \in A$.

5. What are the cardinalities of $A \times B$, $B \times A$, and $A \times B \times C$?

6. What are the elements of $P(B)$?

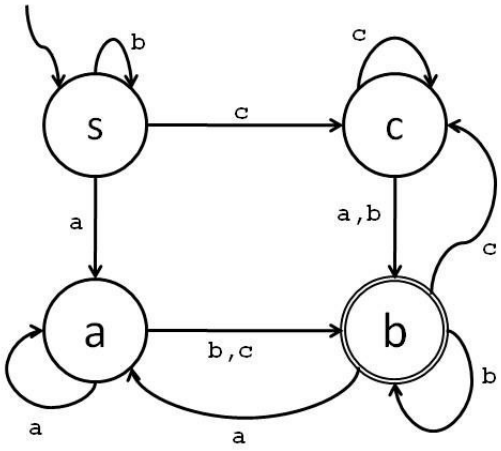
7. What is the cardinality of $P(A \times B \times C)$? (You may leave this as a mathematical expression as long as simply plugging it into a scientific calculator would yield the correct results.)

8. Give a function mapping A to $P(B)$ whose inverse exists (i.e., is also a function) or explain why no such function exists. You may use an arrow diagram or any other clear method to illustrate your function.

[14] The exam may also have a question focused on execution of set operations on particular sets.

11 DFAs and Sequential Circuits [SOME marks][15]

Consider the following DFA:



1. Write out the sequence of states that the DFA would move through if given the input $bbacbccba$. (The first state in your sequence should be the start state; the last should be the one the DFA ends in after reading the entire string.)

2. Give two different strings of length ten that this DFA accepts.

3. Try to describe in English the language that this DFA accepts. (Note: we sometimes ask you to describe a DFA's language in English when there's a pithy English explanation. In this case, there doesn't seem to be any succinct explanation, but for practice, put together a few and debug them to see what's wrong with them! If you find one that works, post it on the bulletin board for an acid test from your peers!)

4. Describe using a regular expression the language that this DFA accepts.

[15] The exam may have a question with a similar format focused on analyzing a DFA and translating it into a sequential circuit.

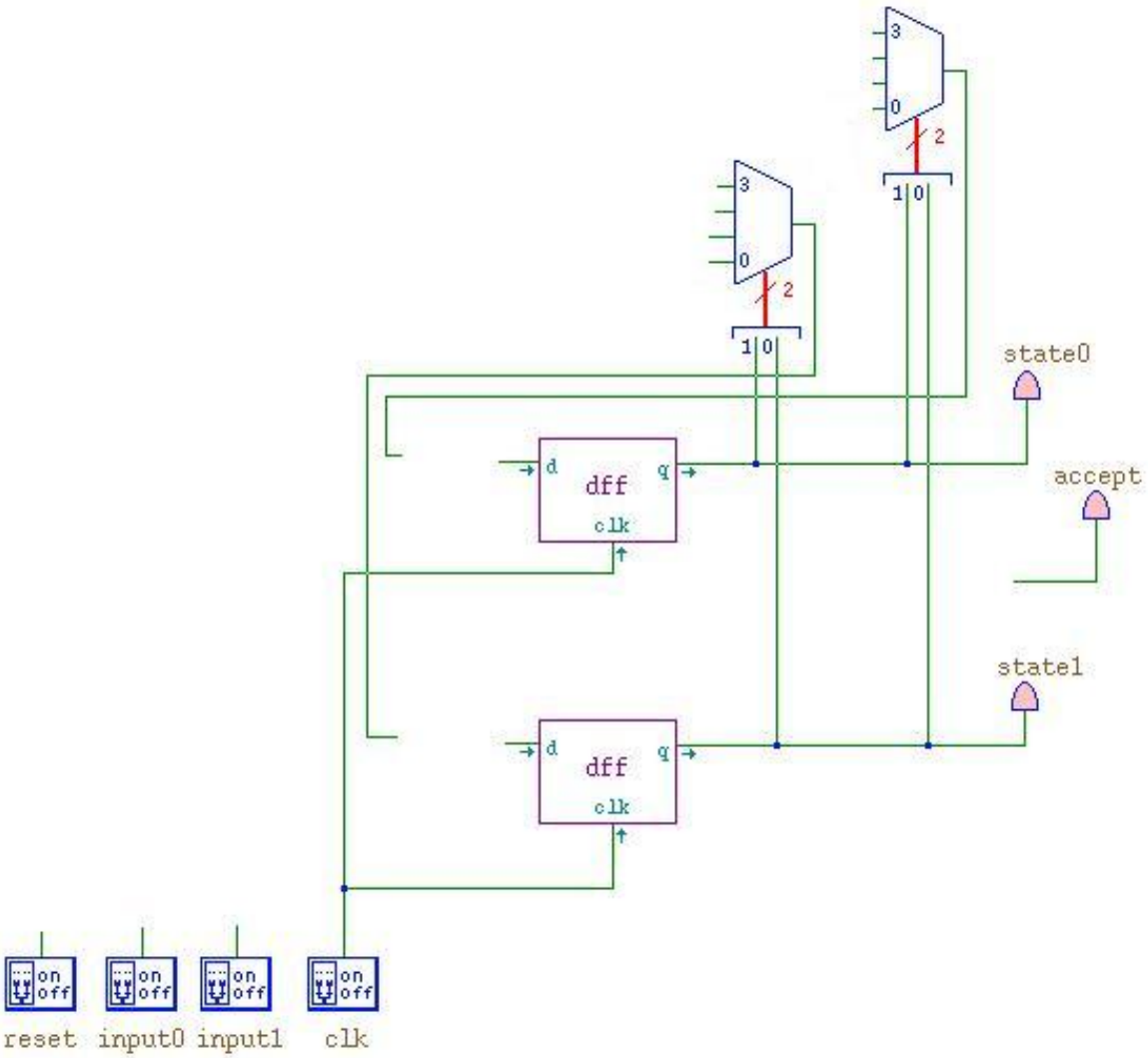
5. Let $a = 00$, $b = 01$, $c = 10$, and $s = 11$. Now, design a circuit using two DFFs that implements this DFA. Your circuit should take the following inputs:

- clk: sent directly to the clock inputs of the two DFFs
- $reset$: when true (on a low-to-high clock transition), the DFA returns to its start state
- $input_0$ and $input_1$: the two bits that together form the input (where, e.g., $input_0 = 0$ and $input_1 = 1$ is 01 and means b). Note that your circuit will never receive input 11.

Your circuit should have three outputs: $state_0$ and $state_1$ (two bits that together indicate the current state, e.g., $state_0 = 1$ and $state_1 = 0$ is 10 and means c) and $accept$ (true exactly when the DFA is in its accepting state).

Your circuit should calculate four separate "next values" for each DFF, one pair for each of the four states the DFA might be in. Pass the four values through two 4:1 multiplexers that use the current state as their control lines, and use the results to update the DFFs' values.

You can start from the following skeleton circuit, which includes all the inputs and outputs, the two 4:1 multiplexers you'll need with their control lines correctly connected, and the $state_0$ and $state_1$ outputs wired up correctly:



12 The Simple Computer [SOME marks][16]

The simple computer we studied is a basic fetch-decode-execute machine. Data flows to and from the major components of the computer using a single, 32-bit wide, data bus. There is at most one source of 32-bit data enabled to the data bus per clock cycle. Four special purpose registers are potential sources/destinations of data on the data bus. They are the program counter (PC) in the PC32 component, an accumulator (ACC) in the ALU component, the instruction register (IR) in the microController component and the memory address register (MAR) in the memory component. In addition, memory (Mem) itself is a possible source/destination of data on the data bus.

The control bus is 16-bits wide. It specifies the source and destination for data flow on the data bus. In addition, it contains signal lines used to specify to each component exactly what operations are required. The microController plays the role of "traffic cop" decoding each instruction to determine the sequence of data transfers on the data bus and to specify the associated control signals so that the intended operations are achieved.

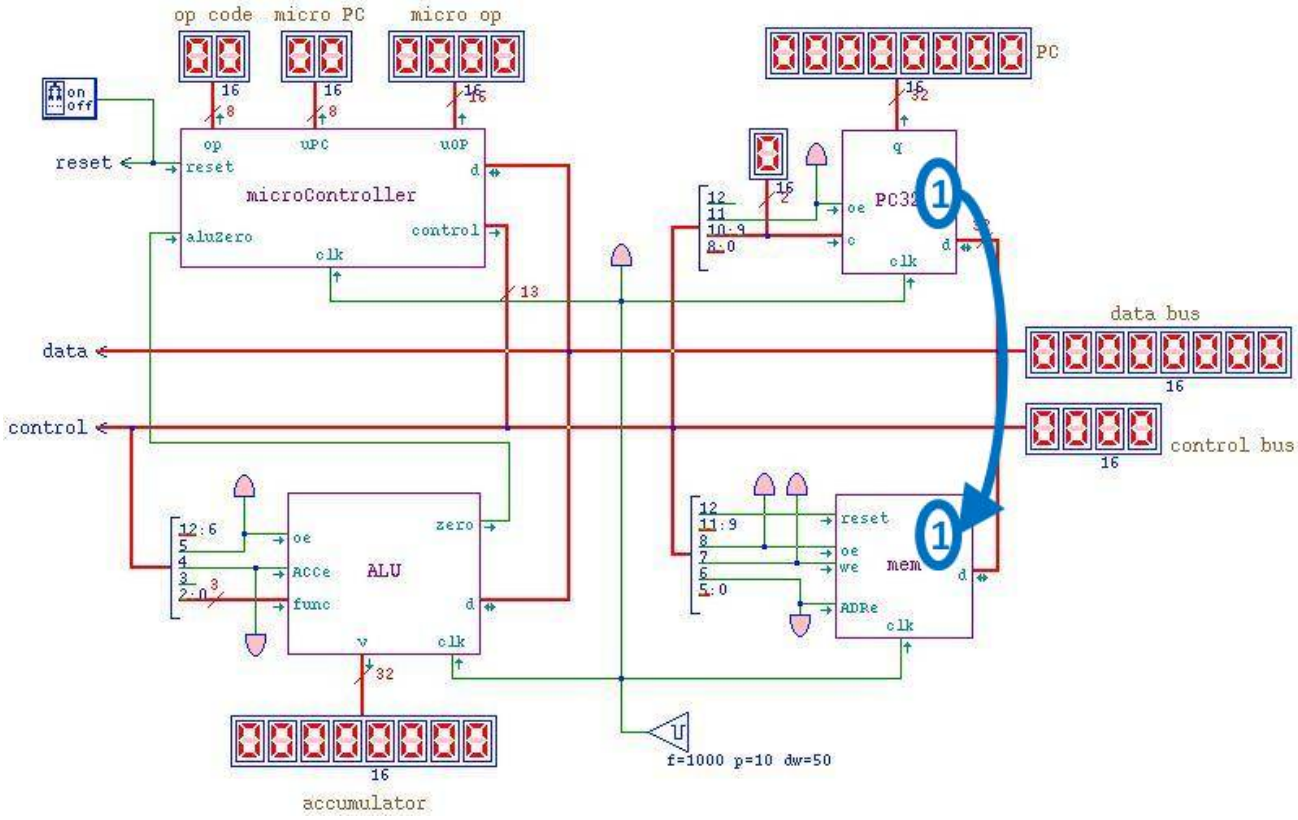
Owing to this shared use of a single data bus, multiple clock cycles are required to complete the fetch-decode-execute cycle for each machine language instruction. Consider the instruction for storing the contents of the ACC to memory. This involves determining the memory address to use and storing the current value of the ACC into that memory location. The complete fetch-decode-execute cycle for a store instruction requires the simple computer to perform four steps, in sequence. Each step uses the data bus.

1. For each step, indicate the source and the destination (i.e. one of PC, ACC, IR, MAR or Mem for each) for the data passed along the data bus at that step. As an example, the first entry, "1. Note the memory address of the next instruction," is already provided for you. The PC is the source (i.e., it is where the memory address comes from) and the MAR in the memory component is the destination (i.e., it is where the address is sent to be "noted" for use in the next step). Add the required entries for steps 2, 3, and 4.

Operation	Source	Destination
1. Note the memory address of the next instruction	PC	MAR
2. Load the instruction stored at that memory address	_____	_____
3. Note the memory address where the data is to be stored	_____	_____
4. Store the value	_____	_____

[16] The exam may have a question with a very similar format. If so, the background text will be similar to the text given here, which can save you some reading time.

2. One “wire” (bit 15) of the control bus is labelled “next instruction.” If set, bit 15 causes the value read from the data bus to be stored in the IR, the Program Counter to be incremented by one, and the “microPC”—which tracks how far along the microcontroller is in the microcode instructions implementing a machine level instruction—to be set to zero. Bit 15 typically is set as the last microcode step in the execution of each instruction. What is the role of each of these three steps in completing an instruction?



This page intentionally left blank.
If you put solutions here or anywhere other than the blank provided for each solution, you must *clearly* indicate which problem the solution goes with and also indicate where the solution is at the designated area for that problem's solution.

This page intentionally left blank.
If you put solutions here or anywhere other than the blank provided for each solution, you must *clearly* indicate which problem the solution goes with and also indicate where the solution is at the designated area for that problem's solution.

This page intentionally left blank.
If you put solutions here or anywhere other than the blank provided for each solution, you must *clearly* indicate which problem the solution goes with and also indicate where the solution is at the designated area for that problem's solution.