EECS 583 – Fall 2023 – Midterm Exam

Friday, November 3, 2023
Exam duration: 1 hr 45 min
Open book, open notes

Name: _____

Please sign indicating that you have upheld the Engineering Honor Code at the University of Michigan.

"I have neither given nor received aid on this examination."

Signature: _____

There are 10 questions divided into 2 sections. The point value for each question is specified with that question. Please show your work unless the answer is obvious. If you need more space, use the back side of the exam sheets.

Part I: Short Answer
5 questions, 25 pts total Score:_____

Part II: Long Answer
5 questions, 75 pts total Score:_____

Total (100 possible): _____

Part I. Short Answer (Questions 1-5) (25 pts)

1) A new compiler identifies possibly uninitialized variables by examining the USE-DEF chains for each variable where an empty chain (e.g., no DEFs that reach the use) indicates an uninitialized variable. Would this method identify *all* uninitialized variables? Briefly explain why or why not. (5 pts)

2) Profile information can be used for a variety of purposes in a compiler. Name a way it can be used to optimize instruction cache performance. Briefly explain. (5 pts)

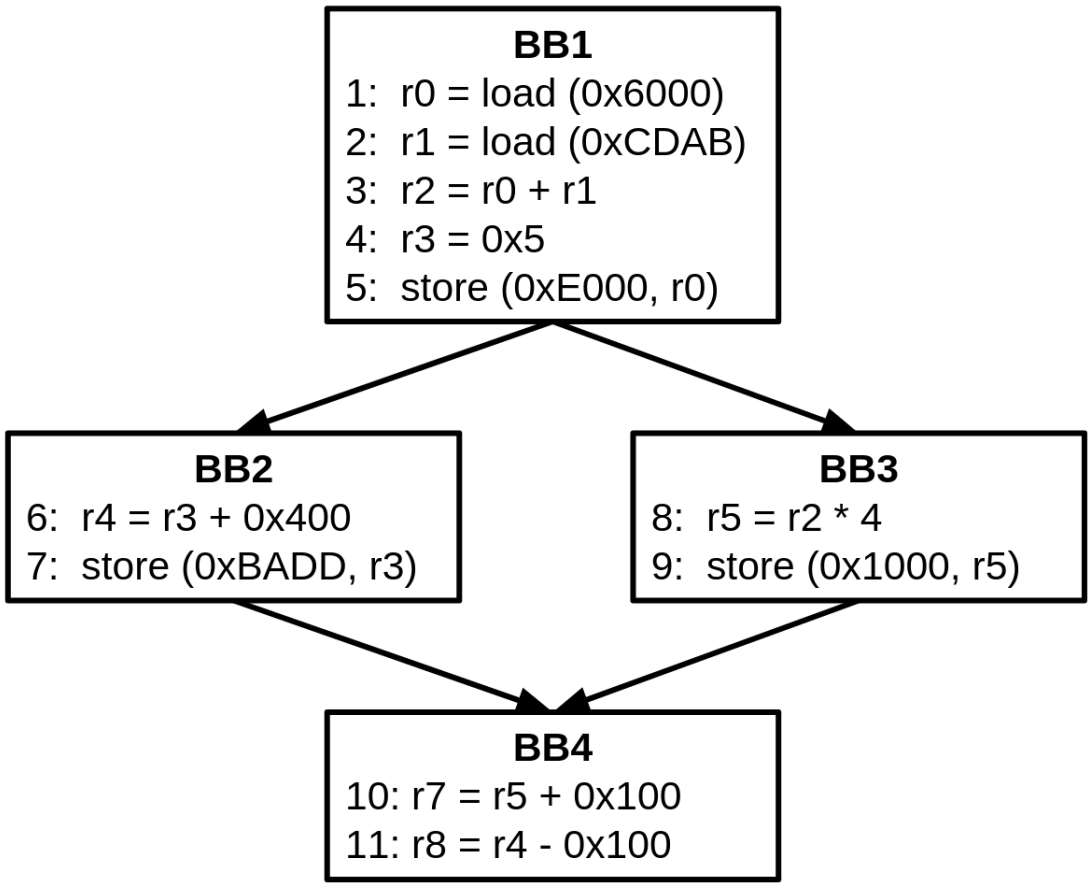
3) When scheduling a basic block, all instructions must be scheduled at their Estart (earliest start) time to ensure the basic block finishes in the fewest cycles. Is this statement True or False? Briefly explain. (5 pts)

4) Is it possible to unroll the following loop, for (i=0; i<100; i+=X) { … } where X is a value input by the user immediately before the for loop statement? *You may assume there are no breaks or continues in the loop and X is not modified*. Briefly explain. (5 pts)

5) For graph coloring based register allocation, all nodes in the interference graph with degree >= N (the number of registers) will be spilled to achieve a successful allocation. Is this statement True or False? Briefly explain. (5 pts)

Part II. Longer Problems (Questions 6-10) (75 pts)

6) You are designing a dataflow analysis for a processor with unreliable memory. Specifically, data in memory between addresses **0xC000 - 0xEFFF** gets lost. Loads from this address range get garbage values and stores to this address range result in data loss. Such loads and stores are considered as faulty. Any subsequent instructions which use the faulty loaded values (arithmetic, memory, or control) are also deemed faulty. Your dataflow analysis needs to identify such faulty instructions. (15 points)
Note: store (A, B) implies that the value B is being stored at address A in memory.
Assume that all registers are initialized properly before BB1 outside the faulty range.



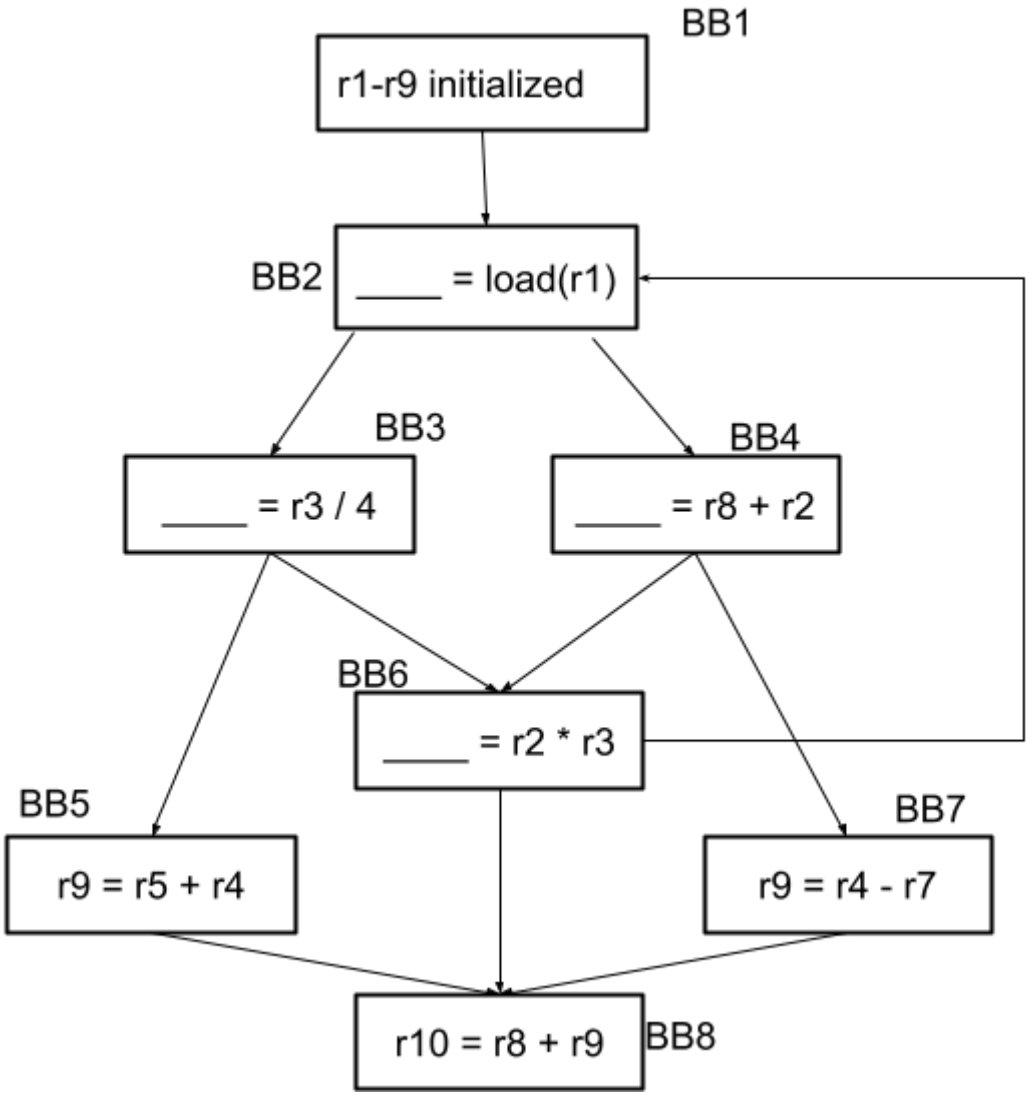
a) Is this a top-down or bottom-up dataflow analysis problem? _____

b) Is this an all-path or any-path dataflow analysis problem? _____

c) Write the GEN and KILL sets for each basic block (just specify the contents of each set for each BB, no need to define the algorithm).

	BB1	BB2	BB3	BB4
GEN				
KILL				

7) Fill in the blanks using r4, r5, r7, and r8, so that a maximum number of instructions from BB2, BB3, BB4, and BB6 become eligible for hoisting via LICM. (10 pts)



8) For the given code, (15 points)

a) Compute the number of predicates required to if-convert the code.

b) To profile the code, we ran this function **100** times, we found that:

i) The loop back-edge was never taken.

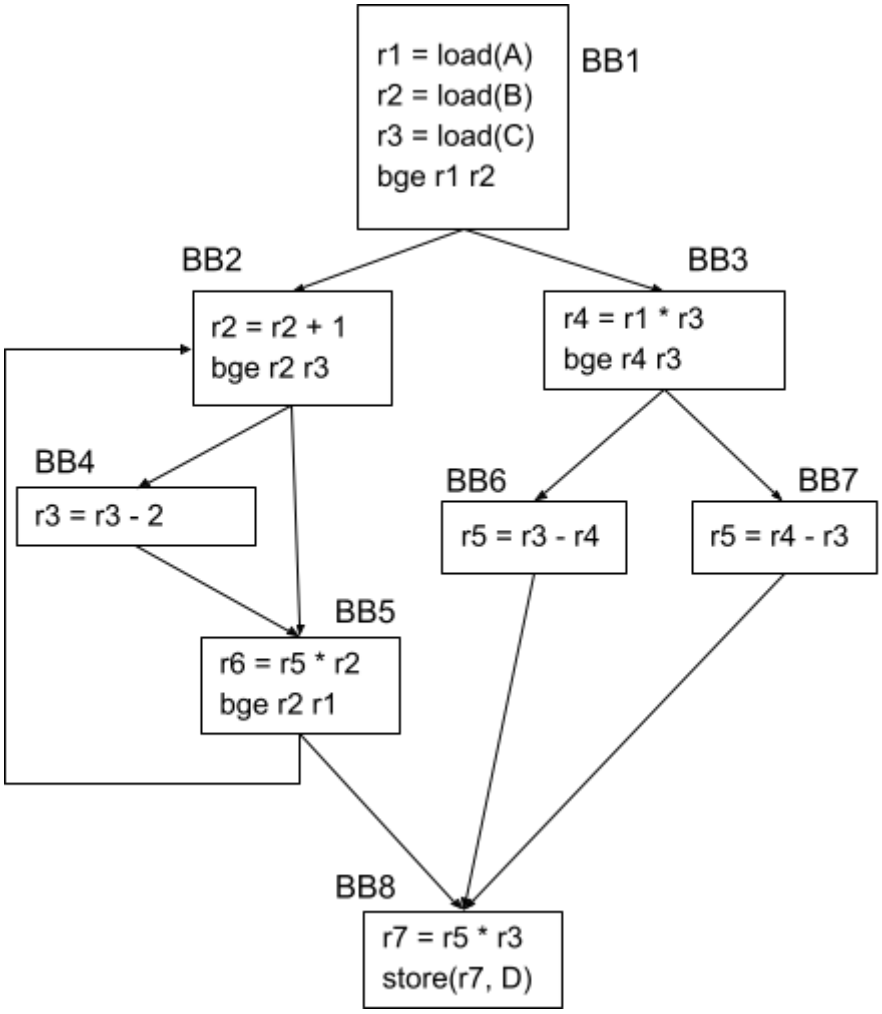
ii) All other conditional branches were taken exactly 50% of the time.

iii) All branch probabilities were independent.

iv) Each instruction takes 1 unit of time to execute, except branches which take 3 units each.

Based on this information, will if-conversion of all eligible branches to utilize predicated execution (no other optimizations) make the code run faster on the same profile test cases? Justify your answer.

You may assume that there are no mispredictions, and that a CMPP instruction can compute up to 2 predicates for every condition.

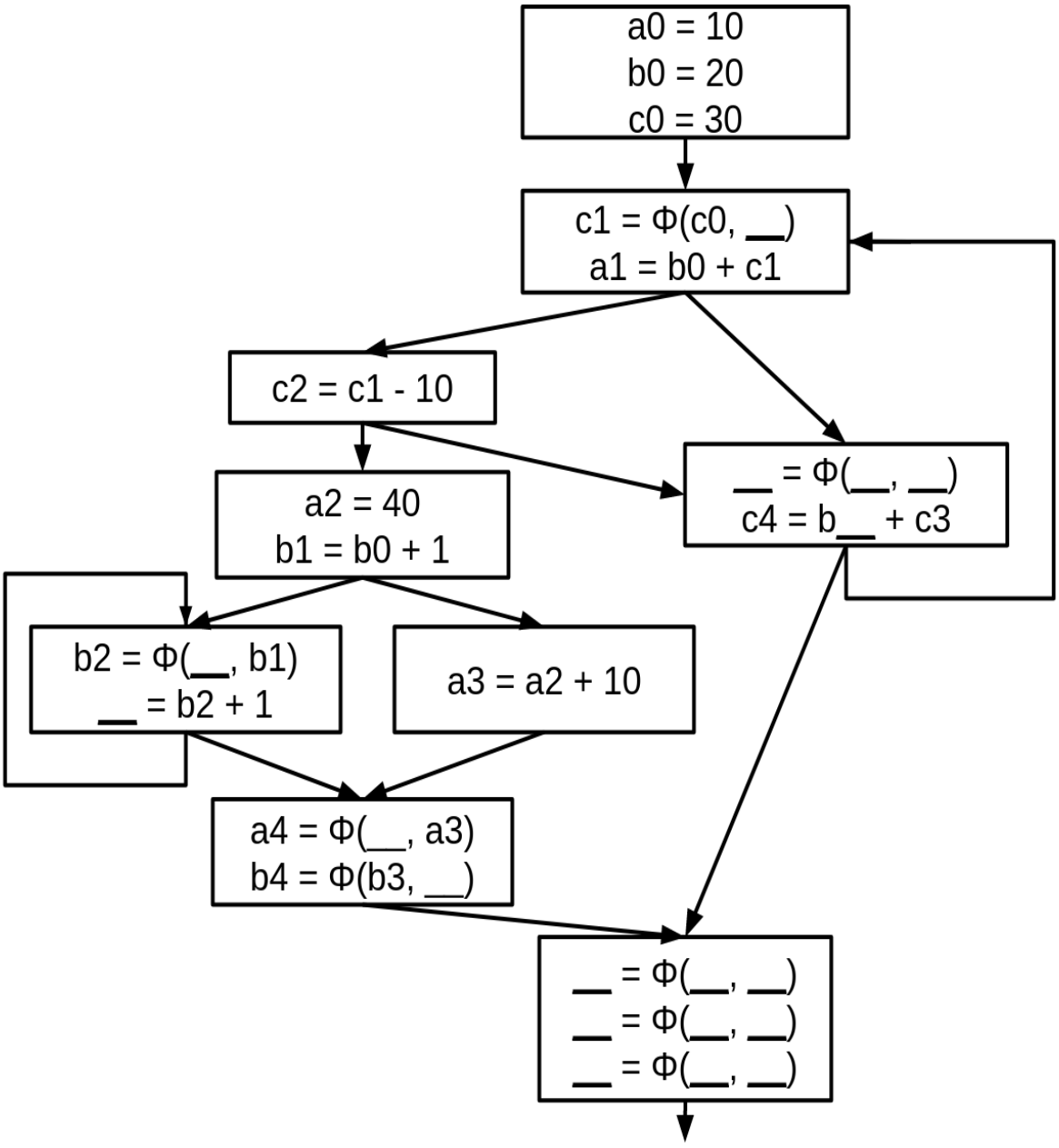


(a) Number of predicates required = _____

(b)

9) Satisfy static single assignment (SSA) form by filling in the blanks in the code segment below. Solving by inspection is fine. (15 points)

- The result and arguments of a phi node must be different instances of the same variable (e.g. a1 = Φ(a2, a3)). Further, all variables with the same letter referred to a single variable in the original program.
- Choose operands between a0 - a5, b0 - b5, c0 - c5. Repetition is allowed.
- There are no unnecessary phi nodes.

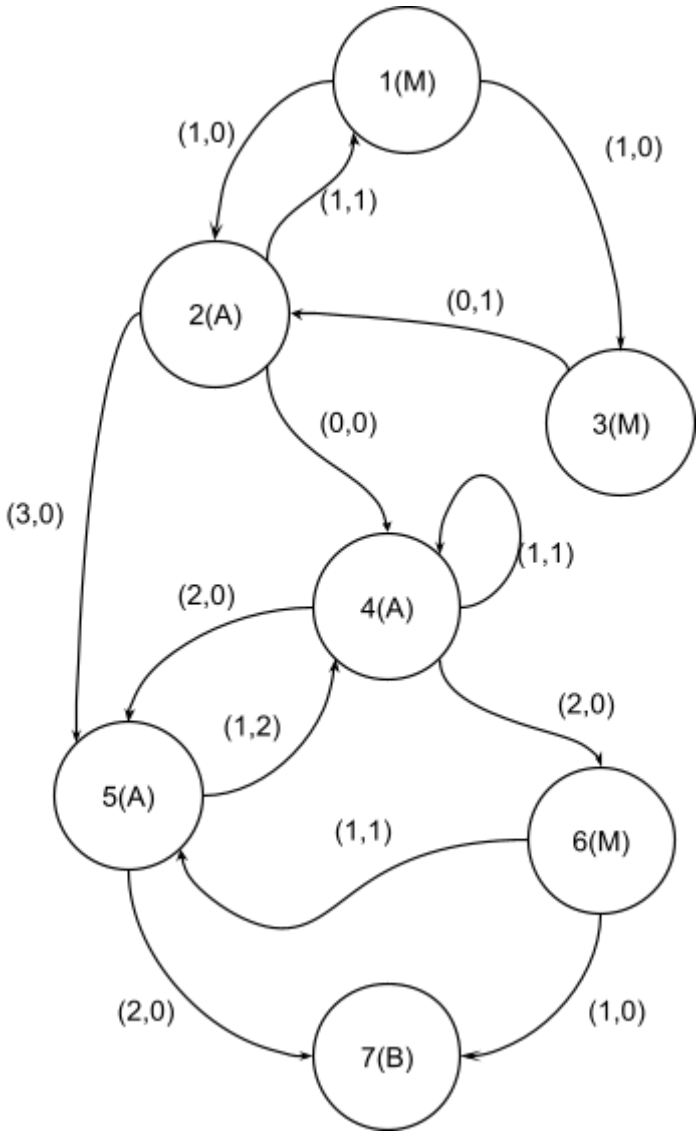


10) Given below is a loop dependence graph and a processor model. (M), (A), and (B) refer to memory, ALU, and branch instructions respectively. The memory instructions use the memory units and the ALU and branch instructions use the ALU units. (20 points)

a. Determine the MII. Show your work.. (5 points)

b. Generate the rolled and unrolled schedules using this MII. Lower instruction numbers will have a higher priority, i.e. instruction 1 has the highest priority. (15 points)

Processor Model
4 fully pipelined function units
– 2 ALU units
– 2 MEM units



Unrolled Schedule (may contain extra rows)

	ALU1	ALU2	MEM1	MEM2
0				
1				
2				
3				
4				
5				
6				
7				
8				

Rolled Schedule (may contain extra rows)

	ALU1	ALU2	MEM1	MEM2
0				
1				
2				
3				

MII = _____