Working copy of Answers

CS251a Midterm Exam 1 (Fall 2018)

Instructions:

- Please answer and hand-in to CCLE by 4:00pm Friday. Make sure you hit submit.
- Please don't put your name on the handin, instead just write your id number. I will try to grade as blindly as possible.
- Please don't discuss questions or answers with anyone.
- Some assumptions are noted, but please feel free to narrow the problem by stating more assumptions, as long as it still basically answers the question.
- For most questions, I listed an expected length of answer, but it's okay to write less or more if you need. There are no points for length.
- If you can use google-docs, write your answers in this document and put your answers in blue. Then just save as a PDF to turn in. Otherwise, you can answer separately, but make sure to answer for each sub-bullet individually.

Problem 1. Anna Karenina Principle

An important principle in computer architecture is balance: sizing or designing structures such that they are balanced with respect to each other, such that none consistently become the limiting factor.

Assumptions: Start with the OOO processor as defined in the slides as the baseline, with perhaps the Pentium 4 or Coffee Lake parameters as "balanced parameters". Assume WIB and CFP are not implemented.

a) Starting with a balanced OOO pipeline, describe what could happen at the pipeline level if the following structures were "too small" or narrow, and *under what program circumstances it would hurt most*. (2-3 sentences each)

- Number of instruction queue (aka. Reservation station) entries

- Number of ROB entries

- Number of Write ports to register file

b) Given other parameters of the pipeline are fixed (sizes and ports on structures), describe a reasonable approach to choosing the physical register file size. (~2 sentences)

Problem 2. Sorting Machines

You are tasked with optimizing the microarchitecture of a core for sorting 32-bit integers. You are told that a wide variety of sorting algorithms are useful, so you should try to support all of them. The inner loops from these kernels are shown below.

Problem assumptions: Please assume N (number of integers to be sorted) is very large. Assume the numbers begin unsorted. Assume each loop below dominates respective execution time.
Compiler assumptions: Assume that the compiler doesn't do loop unrolling or predication. Assume the compiler is fixed, and can't be improved.

Merge from Merge Sort (same as hw1)	Histogram from Radix Sort	Iterative swap from Insertion Sort

`while ((left <= left_end) &&`
`  (mid <= right)) {`
`  if (numbers[left] <= numbers[mid]){`
`  temp[tmp_pos++]=numbers[left++];`
`  } else {`
`  temp[tmp_pos++]=numbers[mid++];`
`  }`
`  }`
	`//Assume Radix 16`
	`//Arr is size N`
	
	`//Count is a histogram of`
	`//hexadecimal digits`
	`for (i = 0; i < N; i++)`
	`  count[ (arr[i]>>shf)%16 ]++;`
		`// i is the index of the candidate`
		`// for swapping`
		`key = arr[i];`
		`j = i-1;`
		`while (j >= 0 && arr[j] > key) {`
		`  arr[j+1] = arr[j];`
		`  j = j-1;`
		`}`

a) Your manager proposes ~~4~~ 3 optimizations. For each, give your best estimate of whether the speedup would be low (<10% speedup) or high (>10% speedup), and also justify by describing which workloads the optimization would have an impact on (if any). (2-5 sentences each)

- Advanced Branch Predictor (vs default simple 2-level in slides)
- Adding a memory dependence predictor (vs either aggressive load ordering or conservative ordering)
- Moving from a superscalar "width" of 2 to a "width" of 3 (fetch/issue/rename/commit/etc)

b) You just hear from the Fab that the die yields are going to be much lower than expected. Which design metric becomes more important from an economic standpoint, and why? (~2 sentences)

Problem 3. Data Tags

Notice how physical register names are used as tags in the R10K wakeup scheme. (1-3 sentences each)

a) Say we modify the simple scoreboard to also use register tags. What would be the consequences (performance/energy/correctness)?

b) Say we modify Tomasulo's algorithm to use register tags. What would be the consequences (performance/energy/correctness)?

c) In general, if we are designing a dataflow (out-of-order) approach, how should we decide what value to use for the dataflow tag?

Problem 4. Dataflow Limit

The dataflow limit defines the maximum performance of a program execution restricted only by true data dependences.

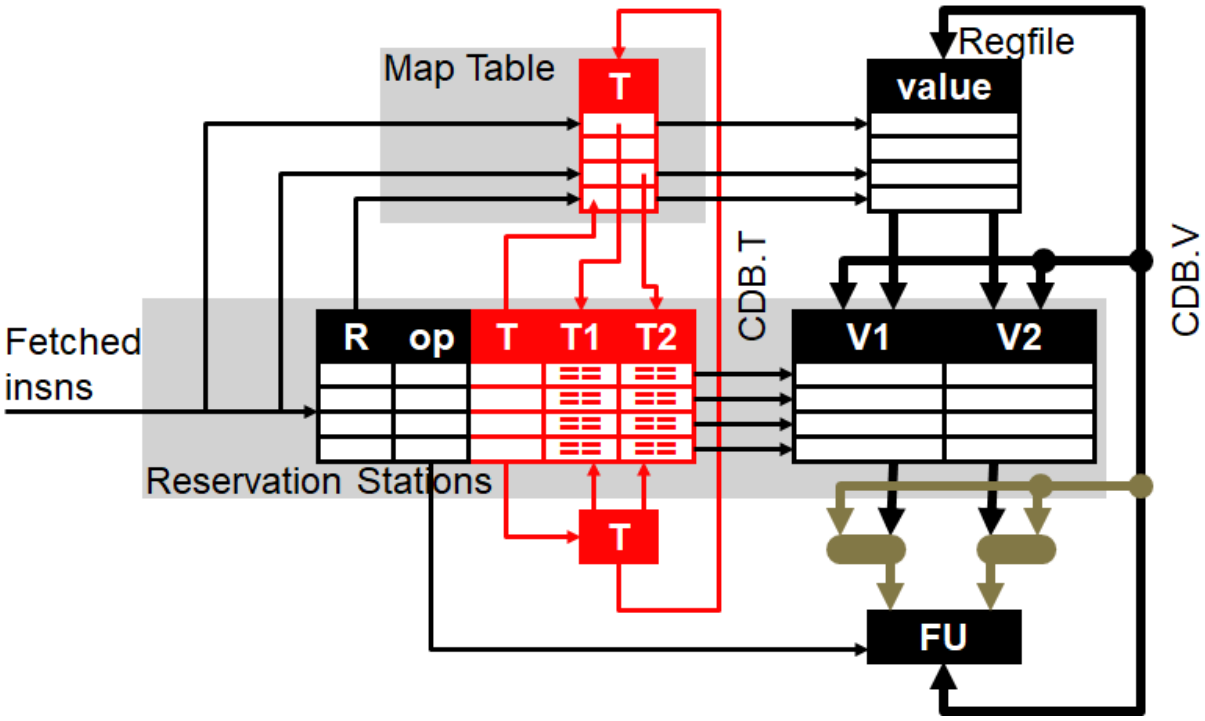
a) Does the dataflow limit depend on the ISA? Why or why not? (2-3 sentences)

b) The dataflow limit is often quantified by a maximum ILP (more precisely max IPC). Give one reason (not involving the ISA) why the estimates of the dataflow limit may have changed over time (ie from 1970s until now). (1-2 sentences)

c) The dataflow limit is sometimes used as an aspirational end-goal for micro-architectural improvements. Explain why you agree or disagree. (3-4 sentences)

Problem 5. Forwarding

Recall our first attempt at adding bypassing to an OOO core, by adding extra forwarding paths from the output of the functional units to the inputs within Tomasulu's design.



	No Bypassing				Bypassing			
Insn	D	S	X	W	D	S	X	W
`ldf X(r1),f1`	c1	c2	c3	c4	c1	c2	c3	c4
`mulf f0,f1,f2`	c2	c4	c5+	c8	c2	c3	c4+	c7
`stf f2,Z(r1)`	c3	c8	c9	c10	c3	c6	c7	c8

a) What was our motivation to do this? Be more specific than performance or correctness. (~2 sentences)

b) Unfortunately, our design at that time was incomplete. Explain the complications. (~2 sentences)

c) Eventually, we talked about a technique that could enable this. (hint: the general form requires a form of speculation) How does it basically work, and what exactly did we need to speculate on? (~2-4 sentences)